TECHNISCHE UNIVERSITÄT MÜNCHEN
Fakultät für Elektrotechnik und Informationstechnik
Professur für Methoden der Signalverarbeitung

Criticality Labeling via Optimal Control and Machine Learning in Automotive Active Safety

Dipl.-Ing. Univ. Stephan Herrmann

Vollständiger Abdruck der von der Fakultät für Elektrotechnik und Informationstechnik der Technischen Universität München zur Erlangung des akademischen Grades eines

Doktor-Ingenieurs

genehmigten Dissertation.

Vorsitzende: Prof. Dr.-Ing. Sandra Hirche
Prüfer der Dissertation:
1. Prof. Dr.-Ing. Wolfgang Utschick
2. Prof. Dr.-Ing. Matthias Althoff

Die Dissertation wurde am 23.04.2018 bei der Technischen Universität München eingereicht und durch die Fakultät für Elektrotechnik und Informationstechnik am 23.01.2019 angenommen.

Für meine Töchter Elisabeth und Charlotte.

Danke.

Diese Dissertation ist im Rahmen meiner Arbeit als Kooperationsdoktorand der *Ingolstadt Institute der Technischen Universität München* (INI.TUM) am Fachgebiet für *Methoden der Signalverarbeitung* im Auftrag der Audi AG entstanden.
Mein aufrichtiger Dank gilt meinem wissenschaftlichen Betreuer Professor Dr.-Ing. Wolfgang Utschick. Durch Detailverliebtheit und viele kreative Anregungen hat er maßgeblich zum Gelingen dieser Arbeit beigetragen und meine Art zu Denken nachhaltig geprägt. Ich hatte bei der Audi AG mehrere kompetente Projektleiter, denen ich danken möchte. Prof. Dr.-Ing. Michael Botsch und Dr. rer. nat. Frank Keck haben das Thema der Doktorarbeit erdacht und die Durchführung im Rahmen von INI.TUM ermöglicht. Dr. rer. nat. Frank Keck hat die Arbeit von Anfang bis Ende hervorragend betreut. Prof. Dr.-Ing. Andreas Gaull und Dr.-Ing. Tobias Dirndorfer haben die Ausrichtung der Arbeit durch ihre Anregungen und Diskussionen geschärft und mir stets ermutigt, eigene Ideen durchzuziehen. Meine Kollegen am Fachgebiet Methoden der Signalverarbeitung waren immer aufgeschlossen, kritisch und konstruktiv; eine bessere Umgebung für eine wissenschaftliche Arbeit kann ich mir schwer vorstellen. Ich danke Dr.-Ing. Christoph Hellings für viele hilfreiche Diskussionen zu Technik und Stilistik. Besonders möchte ich Thomas Wiese erwähnen, der die Arbeit durch einige wichtige Hinweise positiv beeinflusst hat. Ich bedanke mich bei den Kollegen der betreuenden Abteilung *Entwicklung PreCrash Funktionen* unter der Leitung von Quirin Sterner und der Abteilung *Erprobung PreCrash Funktionen*. Es herrschte immer eine offene und intensive Atmosphäre und die Kollegen waren stets bereits, Erkenntnisse aus der Vor- und Serienentwicklung zu diskutieren. Ich erinnere mich gerne an viele Praxistipps und aufschlussreiche Diskussionen mit Christian Wendel, Stefan Katzenbogen, Werner Niedermeier und Dr.-Ing. Patrick Gräbel.

Das Wichtigste für mich ist meine Familie. Ich danke meinen Eltern Marina und Wolfgang Herrmann für ihre Liebe und bedingungslose Unterstützung. Besonders danke ich meiner Frau Anne, die mir mit ihrem Humor, ihrer Liebe und Freude immer wieder Energie gibt.

Stephan Herrmann, München, den 29. Oktober 2017

Zusammenfassung

Die Kritikalität ist ein zentrales Merkmal der Situationsinterpretation für Systeme der aktiven Sicherheit und des autonomen Fahrens. Sie beschreibt die Gefahr der aktuellen Fahrsituation und dient als Entscheidungskriterium für Sicherheitsfunktionen, wie zum Beispiel Gefahrenwarnungen oder das automatische Notbremsen und Ausweichen. Für die virtuelle Entwicklung von Situationsinterpretationsalgorithmen werden Referenzlabels der Kritikalität benötigt. Diese Labels können als Vergleichsgrößen verwendet werden um die Performance von echtzeitfähigen Algorithmen zu bewerten, sowie zum Trainieren von Algorithmen, die auf Machine Learning basieren. Um eine Referenzkritikalität für Auffahrsituationen zu berechnen, wird in dieser Arbeit ein Optimalsteuerungsproblem (OSP) für kombinierte Brems-Lenk-Manöver formuliert und numerisch gelöst. Die Kritikalität der Fahrsituation ergibt sich aus dem minimierten Maximalwert der Beschleunigung der Vermeidungstrajektorie. Die Problemformulierung berücksichtigt die Kollisionsvermeidung für bewegte Hindernisse, sowie gekrümmte Spurbegrenzungen. Als Fahrdynamikmodell dient ein nichtlineares Einspurmodell. Im zweiten Teil der Arbeit wird ein Verfahren vorgestellt, in dem ein echtzeitfähiger Kritikalitätsschätzer auf Basis der Referenzwerte trainiert wird. Hierzu wird das OSP für eine große Anzahl stochastisch erzeugter Kollisionsszenarien gelöst, um deren Kritikalität zu bewerten. Diese Kritikalitäten, zusammen mit den Parametern der Fahrsituation, dienen als Trainingsdatensatz für die Regressionsanalyse mittels Random Forests. Die Prädiktionsgüte des Random Forest wird durch Merkmalsselektion deutlich verbessert. Die Eigenschaften des OSP und die Güte des Kritikalitätsschätzers werden in simulierten Kollisionsszenarien untersucht. Zur weiteren Validierung wird die Methodik auf Realdaten eines aufgezeichneten Vermeidungsmanövers angewendet. Die Simulationen demonstrieren die Eignung der numerischen Optimalsteuerung für die Erzeugung von Referenztrajektorien und -Kritikalitäten für Brems-Lenk-Manöver. Random Forest-Regression erweist sich in den simulierten Szenarien als zuverlässige und rechenzeiteffiziente Methode der Kritikalitätsschätzung. In allen Auswertungen werden die Limitationen der Optimalsteuerungs- und Lernverfahren herausgearbeitet.

Abstract

Criticality is a key metric in situation interpretation algorithms for active safety and self-driving systems. It describes the threat of the current driving situation and serves as a decision criterion for safety functions, e.g., warning the driver and automatic emergency braking and steering. Simulation studies for the development and validation of situation interpretation algorithms require labels of the ground-truth criticality of a driving situation. These labels can be used as a reference to quantify and compare the performance of real-time algorithms and for the training of situation interpretation algorithms based on machine learning. To find such a ground-truth criticality for rear-end collision avoidance, we propose an optimal control problem (OCP) formulation of criticality labeling for a prediction horizon of a few seconds. We minimize the maximum-norm of the acceleration of the avoidance trajectory. The OCP formulation considers lane-keeping constraints for clothoidal lanes, a nonlinear single-track model for the host vehicle and anti-collision constraints. Next, we present a labeling and learning approach to real-time criticality estimation. We numerically solve the criticality OCP for stochastically generated critical driving scenes to obtain a large labeled dataset. Using Random Forest regression with supervised feature selection, we harness this dataset to find a set of features that best predict the criticality of the driving scenes. The performance of the OCP labeling algorithm and the Random Forest regression are evaluated on simulated collision scenarios and in a resimulation of a driving experiment on a test track. Our experiments demonstrate that our OCP formulation yields quality, interpretable criticality labels and evasion trajectories and that Random Forest regression reliably predicts the criticality labels. We analyze shortcomings and extensions of the labeling approach.

Contents

Chapter 1

Introduction

The safety and comfort of modern series production cars have been improved significantly through the introduction of so-called active safety systems (ASS). For example, in the United States, Forward Collision Warning and Automatic Emergency Braking systems have led to a reduction of police-reported front-to-rear crash rates by around 40 percent [1]. An ASS measures the environment of the host vehicle[1] using sensors like video and radar, analyzes the criticality of the situation, and prevents accidents, e.g., by warning the driver or by braking automatically. Criticality is a measure of the threat inherent in a driving situation. It is a vital component of any situation interpretation algorithm for ASS and self-driving.

In this work, we propose a criticality labeling and learning approach where the criticality of a driving scene is determined by computing optimal evasion trajectories. First, we derive an optimal control problem (OCP) for computing physically-motivated ground-truth labels of the criticality of collision avoidance maneuvers by braking and steering. While we can elegantly encode the available expert knowledge about collision avoidance in the OCP, it is not suitable for real-time application due to the high computational complexity of numerically solving the OCP. To obtain a computationally efficient criticality estimator, we generate a dataset of collision scenarios and OCP criticality labels, which we exploit using supervised machine learning techniques. We quantify and validate the performance of the criticality estimator by comparing against the OCP labels on synthetic and real-world data. In the following, we introduce the required terminology, prior art, and contributions.

[1] The host vehicle refers to the vehicle in which the active safety system is operating. It is also referred to as the EGO vehicle.

1.1 Automotive Active Safety Systems

The goal of active safety systems (ASS) is to either completely prevent an imminent collision or to reduce the collision impact. In contrast, passive safety systems, e.g., airbags and seat belts, decrease the risk of injury for passengers and traffic participants after the time of impact of a collision. The main components of an ASS are the environment perception, situation interpretation, and actuation.

Environment Perception Environment perception is the signal processing step that creates a model of the environment from the raw sensor data of the exteroceptive sensors. This environment model provides all information necessary for the situation interpretation of the safety function. Core tasks in environment perception are object detection and classification as well as the estimation and tracking of the kinematic state of moving objects[2].

A critical part of environment perception is the detection and classification of static and moving obstacles. The detection of objects in visual images roughly consists of two steps, feature extraction and classification. Typical methods to compute a feature vector from image patches are, e.g., the histogram-of-oriented-gradients [2] and the Scale-invariant feature transform [3]. The feature vectors are computed for a large training set of images and object labels for training supervised machine learning classifiers (e.g., Support-Vector Networks [4], AdaBoost [5]). The classifiers detect whether an image or image patch contains an object and to which object class (e.g., pedestrian, car, tree) it belongs. An extensive survey and comparison of pedestrian detection algorithms for monocular images is provided by [6], [7] and [8]. With radar sensors, the disambiguation between object types relies on tracking and classifying the kinematics of the object [9]. State-of-the-art object recognition relies on Deep Learning [10] where large neural networks take the raw image pixel data as input vectors to detect the presence of objects. The Deep Learning approach does not explicitly separate between the feature extraction and classification steps. Instead, feature extraction is part of the neural network layers and is therefore adapted to the available data during the training phase of the deep network.

The detection and tracking of lane markings has enabled driver assistance systems like Lane-Departure-Warning and Lane Keeping [11]. In [12], two multi-focal cameras towards the front and rear of the test vehicle were used to track lane markings modeled as segments of clothoids (a type of spiral). This vision system was central for the first prototypical demonstration of autonomous driving, *VaMoRs-P* [13], on a German Autobahn. For the detection and tracking of 2D lane markings using mono video and

[2]The distinction of the discrete steps *detection*, *classification*, and *tracking* in this overview merely serves as a guide and can blur significantly in complex real-world systems.

particle filters, see [14, 15]. In [16], an extension to a 3D representation of the lane is presented that takes the height profile of the lane into account.

Another significant part of environment perception is the estimation and tracking of the state of the environment. Adaptive cruise control systems, as well as some pre-crash collision avoidance systems, rely on millimeter-wave radar [17, 18] to detect and track other vehicles. The distance of a reflection point is determined using frequency-modulated continuous waveforms, where the change in frequency between the transmitted and received signal allows to compute the time-of-flight of the wave. For moving obstacles, the Doppler shift of the frequency is also part of the total frequency shift between transmitted and received signal and is used to measure the relative velocity of the EGO vehicle and the obstacle. Multiple frequency sweeps can be used to determine the distance and speed of the obstacle at the same time [18].

State-of-the-art ASS use sensor fusion of radar, video, and lidar (light detection and ranging) data. The tracking and fusion of objects, as well as the estimation of different types of occupancy maps, are treated in depth in [19] and [20]. Occupancy maps provide information about which parts of the environment can be traversed without collision. For vision-based environment perception, [21] surveys the literature on pedestrian detection comparing preprocessing, segmentation, and tracking. A survey of vision-based vehicle detection, tracking, and behavior analysis can be found in [9]. While radar and camera-based systems currently dominate the commercial market for driver assistance and safety systems, lidar technology is widely used in research-grade autonomous driving [22], and first series production cars with lidar are announced for 2017 and 2018, e.g., the 2017 Audi A8. Lidar measurements are often fused with camera images to aid with object classification, see, e.g. [23].

Situation Interpretation and Criticality Measures The task of analyzing the available information about the EGO vehicle and the environment, to detect imminent collision threats, and to trigger suitable counter-measures is commonly referred to as *situation interpretation* (SI). The SI requires information about the current state of the EGO vehicle and the environment perception, e.g., other dynamic and static objects, lane markings, and occupancy grids. Based on these inputs, SI algorithms predict the future trajectory of the host vehicle and obstacles in the environment with a prediction horizon of a few seconds. More advanced ASS, as well as autonomous driving functions, also compute evasion trajectories that keep the EGO vehicle on a collision-free path. The prediction of the driving scenario is the basis for estimating the threat or *criticality* of the driving scenario. The criticality is a key metric for choosing an appropriate safety intervention. A higher value of the criticality should correspond to a higher threat inherent to the driving situation which in turn requires a more severe safety intervention from a warning to emergency braking.

A significant challenge of criticality estimation is that the criticality is not an

objective, measurable quantity. Thus, there exists a multitude of definitions of criticality and diverse algorithmic approaches to its computation[3]. We briefly survey three common, physics-based criticality metrics, i.e.,

- time-to-X measures, which indicate the time left to a critical event,
- the probability of collision,
- and the intensity of avoidance maneuvers.

For further reading on criticality measures and their computation, we recommend the dissertation [26, Ch. 5.3] and the more recent survey paper [27].

Time reserves measure the time left until a relevant event happens if the driving scenario continues as predicted. In adaptive cruise control systems, a typical time reserve is the headway time, which is the time until the host vehicle has traversed the distance to the vehicle in the front [28]. While the headway time is a useful measure of the potential of traffic conflicts, it is not suitable for detecting imminent collision threats. Imminent collision threats are better detected by the so-called time-to-collision [29]. Another version of time reserves are the time-to-react measures [30], [31]. It measures the time that is left before the collision becomes unavoidable by braking, steering, or a combination thereof.

The probability of collision is a criticality measure that takes into account the uncertainty of the prediction of the traffic scenario. To this end, a priori-assumptions are made about the probability distributions of the state estimate of the host vehicle and the environment and the future actions of the traffic participants. For simple kinematic models, Gaussian probability distributions, and neglecting road boundaries, the probability density functions of the traffic participants can be propagated forward in time in closed form [32]. A more common and flexible approach is the Monte Carlo sampling of future trajectories as used, e.g., in [33], [34], [35]. There, random samples are generated from the probability distributions of the state estimates and future actions. For each sample, the trajectories of the host vehicle and the environment are simulated and checked for collisions; the collision probability is the fraction of samples that result in a collision. The Monte Carlo method allows incorporating nonlinear vehicle dynamics as well as road boundaries, albeit at an increased computational cost proportional to the number of simulated trajectory samples. Probabilistic reachability analysis has been used in [36], [37] to compute collision probabilities of trajectory plans. This method computes so-called reachable sets for future time instants by over-approximating the set of states and the possible actions of the host and obstacle

[3]The subjectivity of the criticality of a driving scene can be taken into account by a driver model, e.g., for lowering activation thresholds in case of driver inactivity. Driver modeling is an emerging field spanning driver attention estimation, position- and pose estimation, intent recognition, and more [24], [25].

vehicles. The probability density of the state and future actions is incorporated such that the reachable sets correspond to an enclosing hull of the density distributions.

A third approach to criticality estimation is to compute the intensity of avoidance maneuvers. The avoidance intensity quantifies the minimum control intervention that is necessary to avoid a collision. In [38, 39, 40] the lateral and longitudinal avoidance accelerations are computed, and the criticality is given by the ratio of avoidance acceleration to the maximum possible acceleration. The maximum acceleration depends on the friction coefficient as well as on the steering or braking system of the host vehicle. An alternative intensity measure is the jerk, i.e., the rate of change of the acceleration as used, e.g., in [41, 42]. The avoidance intensity approach requires assumptions about the future behavior of the obstacle vehicle. Compared to the probability of collision criticality metric, we can compute the avoidance intensity without specifying probability distributions for the state and action of host and obstacle vehicle.

Actuators for Active Safety Depending on the state of the host vehicle, the driver behavior, and the criticality estimate, an Active Safety System can trigger different actors to warn the driver of the impending collision or to mitigate the collision automatically. In the case of an impending collision, the warning of the driver by haptic feedback of the steering wheel or by sound can allow them to avoid the collision on their own. Pre-filling the brakes reduces the dead time, and the reversible tightening of the seat belts can reduce the injury risk in case of a crash. The risk of secondary collisions can be reduced by flashing the tail lights or warning flasher. If the criticality is very high, i.e., the collision is almost inevitable, Automatic Emergency Braking (AEB, see, e.g., [39]) automatically triggers the brakes. Slowly, systems for automatic steering interventions are coming to market.

While this work concentrates on braking and steering, there are many more actuators that can help avoid a collision or mitigate its severity. The entire Human-Machine-Interface between the host vehicle and the driver can and should be considered a safety-critical actuator. This is highlighted by the hand-over-phase in conditional autonomous driving (SAE Level 3, see, [43]). Conditional automation will allow the driver to concentrate on tasks other than driving or monitoring the road, e.g., watching a video, surfing the internet, or turning towards other passengers. In case of a disengagement of a level 3 system, the driver still has to take control of the vehicle within a time interval in the range of ten to 30 seconds (for now, there is no industry standard for this time interval). If the driver does not take over control of the car immediately, the entire car interior could be engaged to get the driver to retake control, e.g., by dynamic interior lighting, sound, moving the steering wheel or using seat position actuators.

1.2 Ground-Truth Criticality Labeling using Optimal Control

In the first chapters of this thesis, we develop a Numerical Optimal Control formulation for labeling the ground-truth criticality of rear-end collision scenarios. Optimal control theory offers an appealing framework for criticality labeling as it allows including a model of vehicle dynamics in terms of a differential equation constraint and facilitates an elegant and convenient modeling of additional constraints in the problem formulation, e.g., lane keeping and collision avoidance. The criticality labels are defined as the minimized maximum-norm of the vehicle dynamics during a prediction horizon of a few seconds, allowing a straightforward interpretability of the labels in terms of the physics of the collision avoidance problem.

When we say that a label is the *ground-truth*, we mean that it uses the most detailed models possible for the available budget of computational resources and engineering effort. Specifically, a reference algorithm or simulation is not constrained by having to run in real-time in an electronic control unit (ECU) in a vehicle, which is often the focus in the literature on situation interpretation algorithms.

Outline In Chapter 2, we define the coordinate systems used for modeling clothoidal lane segments and introduce the nonlinear single-track vehicle model for the host vehicle as well as its tire models. We also define the model of the driving scenario which comprises the state of the host vehicle and obstacle, the road geometry, and simple assumptions for the prediction of the movement of the obstacle. Chapter 3 reviews the basics of numerical optimal control using the direct multiple shooting [44] (DMS) method. DMS is a state-of-the-art approach to the numerical solution of optimal control problems (OCP) that proceeds in two steps: first, the OCP is transformed to a finite dimensional optimization problem by solving the differential equations on time-intervals with constant control inputs. Then, the optimization problem is solved using numerical optimization solvers. In Chapter 4, we formulate the optimal control problem for the criticality labeling of rear-end collision avoidance maneuvers with braking and steering using the scenario model defined in Chapter 2. To this end, we derive the constraint formulations for lane keeping, collision avoidance, as well as the initial and final state constraints. We propose a novel formulation for lane keeping that allows to model curvilinear roads by incorporating the parameterization of the road boundaries as control variables in the optimization problem. We formulate the collision avoidance constraint in terms of finding a separating hyperplane between convex polytopes. While separating hyperplanes have been used in the robotics literature to detect collisions, see, e.g., an early application in [45], to the best knowledge of the author, this work presents the first application of the separating hyperplane as an optimization variable in an optimal control collision avoidance problem. This general formulation allows modeling moving obstacles

with convex shapes. The design of the cost function and constraints allows a trade-off between the interpretability and feasibility of the optimal control problem (ref., Chapter 4.5.1). Chapter 5 details the concrete application of the DMS method to the problem of criticality labeling. It includes the finite-dimensional approximation of the OCP, the initial guess for all optimization variables, and the regularization of the cost function. We also compare the effect of the choice of the cost-function and tire models on the interpretability of the criticality labels and the ability to find a feasible solution to the OCP approximation using example scenarios.

Related Work and Use Cases Optimal control has been used for collision avoidance algorithms in several previous works, e.g., [46], [47], [48], [49], [50]. While being methodologically comparable, we contribute a problem formulation that is more general regarding the lane and obstacle model and that yields labels that are more interpretable. We refer to these works throughout the appropriate sections of Chapter 4.

Reference datasets with ground-truth labels can be used in multiple contexts. First, they enable quantifying the performance of real-time capable algorithms when confronted with the driving scenarios in the dataset. Comparing the ground-truth criticality to the behavior of the algorithm under test allows computing a performance score. In [51], naturalistic driving data recorded from 15 drivers is partitioned into *threatening* and *safe* situations based on thresholds on signals like braking level and time-to-impact. These labels are then used to score and compare the performance of a variety of collision warning and avoidance algorithms. The authors of [52] propose a benchmark for motion planning algorithms that consists of driving scenarios, score functions, and vehicle models. Such benchmarks could eventually serve the active safety and autonomous driving communities for transparently comparing and improving algorithms. This would compare to the way that labeled image datasets like MNIST [53] (recognition of handwritten digits) or ImageNet [54] (object recognition) have fostered competition and improvement in the area of image recognition.

Second, such performance scores can be optimized to find parameterizations of real-time capable algorithms, see, e.g., [30, 55]. In [30], stochastic simulations are used for finding proper activation thresholds for a collision mitigation algorithm, considering the uncertainty of the behavior prediction of an obstacle. The performance of a parameterization is measured in terms of the gained reaction time and decreased collision velocities, as well as false alarms for stochastically generated collision scenes. In a similar vein, the authors of [55] formulate the problem of finding activation thresholds for a collision warning system as maximizing the trade-off between false-positive and false negative activations in simulation data.

The third use case for driving datasets with ground-truth criticality labels is the training of machine learning algorithms for situation interpretation. This is the use case we concentrate on in the later chapters of this work.

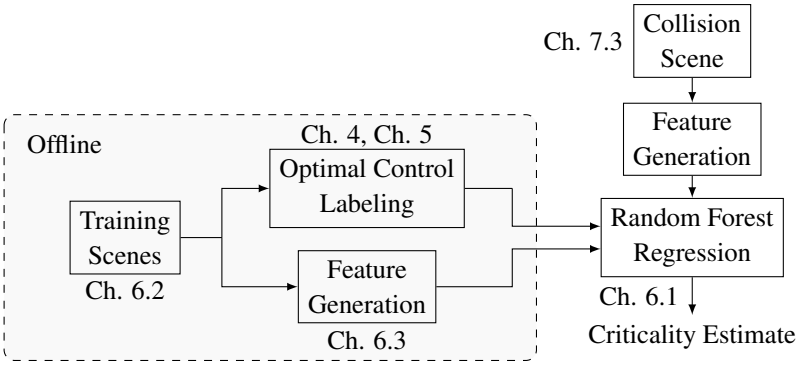


Figure 1.1: Overview of supervised learning and prediction of criticality

1.3 Machine Learning for Real-Time Criticality Estimation

In the second part of this thesis, the goal is to train a criticality estimator using supervised machine learning that closely matches the optimal control-based criticality labels. The criticality estimator profits from the model detail devoted to the optimal control model while being computationally efficient, which allows it to be used in real-time applications, e.g., on an electronic control unit.

Outline Figure 1.1 outlines the main components of the supervised training of the criticality estimator. We train a Random Forest (RF) [56] for regression on a dataset of driving scenes and associated criticality labels. This training is done offline, i.e., before the trained RF is used to predict the criticality of new driving scenes.

For the generation of a large set of rear-end collision scenes, we propose a stochastic model of the state of the host vehicle and the obstacle as well as the lane geometry, see Chapter 6.2. This model allows us to simulate critical driving scenarios with realistic lane geometries and dynamics. We label the criticality of the generated driving scenes by solving the collision avoidance optimal control problem as detailed in Chapters 4 and 5.

Chapter 6.1 recapitulates the theoretical basis of Random Forests and their application to nonlinear regression. Apart from their empirically good performance, Random Forests have multiple desirable properties that make them a suitable choice for the criticality regression, e.g., their robustness to parameter settings and fast training time. When many candidate features are available, Random Forests provide a very natural way of ranking the importance of those features and performing feature selection.

The performance of the criticality estimator can be improved significantly by generating and selecting informative features from the driving scenario as detailed in Chapters 6.3 and 6.4. These features, e.g., avoidance accelerations or time-to-collision, are nonlinear transformations of the quantities that define the scenario. To decide

which features to use for training and prediction, we use supervised feature selection.

We quantify the performance of the criticality prediction techniques in Chapter 7 in terms of the generalization error, i.e., the prediction error of the estimator on driving scenes that were not part of the training dataset. While this approach to quantifying the quality of estimators is typical in the supervised learning community, it is far less frequent in the area of situation interpretation for vehicle safety and illustrates the usefulness of automatic ground-truth criticality labeling which enables this type of performance analysis. We validate the real-world applicability of the proposed methods using a resimulation of a collision avoidance maneuver on a test track. We demonstrate that the scene model can be derived from series production sensors and that the labeling and prediction techniques produce reliable and intuitive results. On the other hand, we identify current shortcomings of the approach which become visible in our experiments.

Related Work and Use Cases Machine learning has been used for the prediction of crash severity of vehicle-to-vehicle or vehicle-to-structure crashes. In this application, the motivation is to obtain a prediction model for the crash severity that is real-time capable, in contrast to finite element model simulations which are time and cost intensive. In [57], a dataset consisting of time-series of chassis acceleration sensors from front-crash experiments was analyzed to derive reliable features for the robust activation of airbags. The reference labels, in that case, are the correct airbag activation times before an unbelted passenger moves forward by a maximum distance. Using logistic regression, the authors of [58] predict the Abbreviated Injury Scale (AIS, a measure of occupant injury), from crash geometry, seat belt status, and occupant posture. In [59], the mechanical impact of a crash with other vehicles and crash barriers is predicted using Decision Trees, Support Vector Machines, and Neural Networks. Both papers use finite element simulations of the vehicle chassis to create the underlying dataset of crash constellations and crash severity measures, with [58] also incorporating a finite element model of the occupant. In [60], the reliability of crash severity estimation during the pre-crash phase is predicted using Random Forest Regression using a dataset generated by traffic simulations and analytical crash models. There, the machine learning model provides a measure of uncertainty of the crash severity prediction which can be used for the activation of occupant restraint actuators.

In the area of collision avoidance, machine learning methods have been employed for subtasks of, e.g., trajectory planning [61], feedback control [62], and the prediction of occupancy grids [63]. The authors of [64] use machine learning to improve the sampling efficiency of a trajectory planner based a Rapidly Exploring Random Tree (RRT, [61]). This reduced sampling complexity is achieved by biasing the trajectory sampling strategy towards promising regions of the state space using a

Deep Convolutional Neural Network (ConvNet). In [63], Deep Neural Networks and Random Forests are combined to predict occupancy grids, i.e., the probability that points around the vehicle will be occupied by an obstacle at a future time. These occupancy grids can be used, e.g., to check if predicted or planned trajectories are collision-free. Neural Networks have also been applied to synthesize feedback controllers for collision avoidance, e.g., [62]. There, the neural network is used to learn a mapping from the state space of the vehicle, the environment, and oncoming traffic to steering and deceleration controls. The training set of optimal avoidance trajectories is derived from the solution of a so-called differential game, where it is assumed that the oncoming vehicle has the objective of colliding with the host vehicle, which leads to worst-case optimal controls.

More recently [65], Deep Neural Networks are applied to self-driving by learning a mapping from raw video data to control inputs from large corpora of recordings of human driving. Proponents of this end-to-end learning approach posit that the task of self-driving is too complex to be modeled as a control task in a state space and should be inferred from raw data instead. Thus, compared to the feedback control in [62], the end-to-end approach skips the state-space representation of the host vehicle and the environment entirely.

Another possible use case for machine learning is data mining, i.e., the analysis of large amounts of recorded driving data for interesting segments. In the context of data mining, the runtime of the data analysis task is proportional to the computational complexity of situation interpretation (SI) algorithm. Supervised learning could be used to replace a computationally demanding algorithm with a fast prediction algorithm comparable to the criticality prediction method proposed in this work.

Note that deploying a machine learning based algorithm in a safety-critical function is very challenging from the perspective of functional safety requirements and testing methodology [66]. A possible solution was proposed in [67] in the form of a complementary use of statistical and analytical algorithms: the statistical algorithm would be responsible for maximizing the field performance of the safety function by minimizing false-positive and false-negative activations. Meanwhile, a physics-based model serves as a validation path for suppressing spurious behavior of the machine learning algorithm.

1.4 Publications and Supervised Theses

We have published parts of this thesis as peer-reviewed conference papers.

- In [68], we give an overview of criticality labeling by optimal control in combination with Random Forest classification. Contrary to this thesis, the criticality labels are discretized in that work into few criticality classes.

- In [69], we compare alternative criticality cost functions regarding their interpretability and of the feasibility of solving the OCP. That publication is the basis of Section 4.5.1.

The author commends Michael Knödlseder for his Bachelor's thesis [70], which treats regression with monotonicity constraints as a way to incorporate a priori-knowledge[4] about the predicted variable, increasing the reliability and generalization of criticality estimators based on machine learning.

1.5 Notation

Vectors and Matrices are written in bold face with vectors as lower case and matrices as upper-case letters, $[\mathbf{a}]_i$ denotes the i-th scalar element of a vector $\mathbf{a}$ and $[\mathbf{A}]_{i,j}$ is the element in the i-th row and j-th column of a matrix $\mathbf{A}$. The all-ones vector is written as $\mathbf{1}$, all-zero vectors and matrices are written as $\mathbf{0}$, and the identity matrix is denoted by $\mathbf{I}$. We write the Euclidean norm of a vector as $\| \bullet \|$ and $\bullet^{\mathrm{T}}$ denotes transposition. Rounding up is written as $\lceil \bullet \rceil$, rounding down is $\lfloor \bullet \rfloor$. Sets other than the real numbers $\mathbb{R}$ and natural numbers $\mathbb{N}$ are typeset in calligraphic upper-case letters, e.g., $\mathcal{A}$. The cardinality of a set is written as $| \bullet |$. A probability density function is written as $\mathrm{p}_X(X = x)$ and abbreviated as $\mathrm{p}_X(x)$. The support of a probability density function is noted as $\mathrm{supp}(\bullet)$. For a discrete random variable X with realization x the probability mass function is written as $\mathrm{P}(x)$. The expected value and variance of a random variable are noted as $\mathrm{E}[\bullet]$ and $\mathrm{Var}(\bullet)$. In formulations of optimization problems subject to constraints, we abbreviate *subject to* as *s.t.*.

[4] For example, criticality is always increasing with the closing velocity between the obstacle and the EGO vehicle - at least in rear-end collision scenarios.

Chapter 2

Modeling of Collision Scenarios and Vehicle Dynamics

In the following, we define a model for the prediction and criticality estimation of rear-end collision scenes that includes a curvilinear road representation and a nonlinear single-track vehicle model. We also introduce the *scene tuple* as the minimal description of the collision scene.

2.1 Coordinate Systems

Clothoids, also referred to as Euler spirals, play an important role for the construction of road segments since their shape leads to a linear increase in the lateral acceleration of a vehicle that drives with constant velocity along the clothoid [71, 72]. This is due to the curvature of the clothoid (i.e., the inverse turning radius) increasing linearly with the arc length l, i.e.,

$$c(l) = l\kappa, \tag{2.1}$$

where κ is the constant rate of change of the curvature. Clothoids have previously been used for tracking and predicting the movement of other vehicles [73], for 3D lane modeling [74, 16], and trajectory planning [75].

In Figure 2.1, we depict a Cartesian coordinate system which has its origin at $l = 0$. The unit vectors are given by the tangent and normal vectors of the clothoid at the origin. For a given rate of change of the curvature, κ, the clothoid path is a

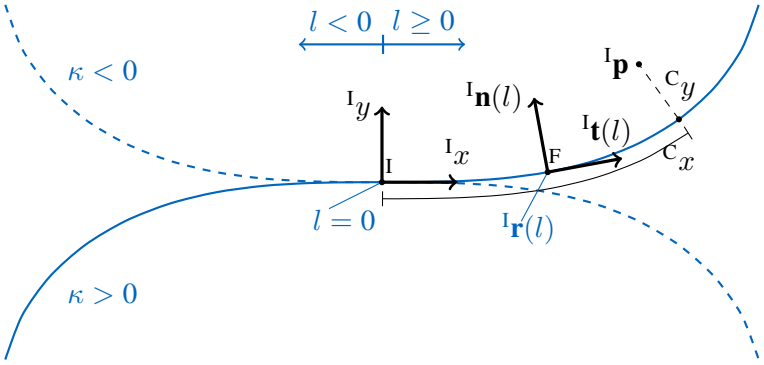


Figure 2.1: Two clothoid segments with positive (blue) and negative (blue, dashed) rates of change of the curvature κ, inertial coordinate system ${}^{\mathrm{I}}$, the Frenet frame ${}^{\mathrm{F}}$, and curvilinear coordinates ${}^{\mathrm{C}}$ of a point ${}^{\mathrm{I}}\mathbf{p}$

function of the arc length l, and is defined by the so-called Fresnel integrals,

$$ {}^{\mathrm{I}}\mathbf{r}(l) = \begin{bmatrix} \int_0^l \cos\left(\frac{\kappa}{2}\xi^2\right) d\xi \\ \int_0^l \sin\left(\frac{\kappa}{2}\xi^2\right) d\xi \end{bmatrix}. \tag{2.2} $$

Since there is no closed form solution for these integrals, in practice, the terms are approximated by a truncated Taylor expansion around $l = 0$ [72]. Consider the depiction of the two clothoids in Figure 2.1. The clothoid which extends to the first and third quadrants (top-right and bottom-left) has a positive rate of change of the curvature κ, while the clothoid in the second and fourth quadrants (top-left and bottom-right) has a negative rate of change of the curvature of $-\kappa$. The clothoid angle ${}^{\mathrm{I}}\tau$, between the unit vector in the x-direction and the tangent vector at the arc length l, is a function of the arc length and rate of change of the curvature, i.e.,

$$ {}^{\mathrm{I}}\tau(l) = 0.5 l^2 \kappa. \tag{2.3} $$

Note that we use the convention that the arc length variable is positive for the clothoid paths in the first and fourth quadrant. We posit that the host vehicle always drives in the direction of an increasing arc length of the lane. Consequently, the four quadrants correspond to a tightening left turn in the first quadrant, a widening left turn in the second quadrant, as well as widening and tightening right turns in the third and fourth quadrants.

Curvilinear coordinates Since we represent lane segments using clothoids, it is convenient to give the coordinates of a point in $\mathbb{R}^2$ in the curvilinear coordinate system (CS) of the clothoid, which we denote by the superscript $^{\mathrm{C}}$. In this CS, ${}^{\mathrm{C}}x$ is the arc length along the clothoid, and ${}^{\mathrm{C}}y$ is the orthogonal complement. Such an aligned CS has previously been used in [73] to express the dynamics of a vehicle that performs lane keeping.

Given a point ${}^{\mathrm{I}}\mathbf{p} \in \mathbb{R}^2$ in the inertial CS (see Figure 2.1), we can compute its coordinates in the curvilinear CS of the clothoid. To this end, we must find a point ${}^{\mathrm{I}}\mathbf{r}(l)$ on the clothoidal path that has the minimal Euclidean distance to ${}^{\mathrm{I}}\mathbf{p}$, i.e.,

$$ {}^{\mathrm{C}}x = \arg\min_{l} \left\| {}^{\mathrm{I}}\mathbf{p} - {}^{\mathrm{I}}\mathbf{r}(l) \right\|_2^2 . \tag{2.4} $$

There are many local optima for the distance of the clothoid path to any point in $\mathbb{R}^2$. Therefore, this optimization is not convex and thus any local solution might not be the global optimum. We alleviate this problem by restricting the tangent angle to the interval $[-0.5\pi, 0.5\pi]$, which is equivalent to a restriction of the absolute value of the arc length. This restriction is not overly conservative concerning realistic lane segments, where the tangent angle of the clothoid typically is smaller than 90 degrees [76, Ch.4.3]. We can now compute the arc length coordinate by solving

$$ {}^{\mathrm{C}}x = \arg\min_{l} \left\| {}^{\mathrm{I}}\mathbf{p} - {}^{\mathrm{I}}\mathbf{r}(l) \right\|_2^2 , \; s.t. \quad l \in \left[-\sqrt{\frac{\pi}{|\kappa|}}, \sqrt{\frac{\pi}{|\kappa|}} \right] , \tag{2.5} $$

where the bounds for the optimization variable result from solving (2.3) for angles $\pm 0.5\pi$. This optimization problem can be solved using standard optimization routines.

To obtain the offset coordinate ${}^{\mathrm{C}}y$, we must project the vector complement from the clothoid path to ${}^{\mathrm{I}}\mathbf{p}$ onto the normal vector at the arc length ${}^{\mathrm{C}}x$, i.e.,

$$ {}^{\mathrm{C}}y = {}^{\mathrm{I}}\mathbf{n}^{\mathrm{T}}(\tau) \left({}^{\mathrm{I}}\mathbf{p} - {}^{\mathrm{I}}\mathbf{r}({}^{\mathrm{C}}x) \right) , \quad {}^{\mathrm{I}}\mathbf{n}^{\mathrm{T}}(\tau) = [\cos(\tau) \sin(\tau)], \tag{2.6} $$

where τ is the tangent angle of the curve evaluated at $l = {}^{\mathrm{C}}x$, cf. Equation (2.3).

The inverse transformation from the aligned CS $({}^{\mathrm{C}}x, {}^{\mathrm{C}}y)$ to the inertial CS is

$$ \begin{bmatrix} {}^{\mathrm{I}}x \\ {}^{\mathrm{I}}y \end{bmatrix} = {}^{\mathrm{I}}\mathbf{r}({}^{\mathrm{C}}x) + {}^{\mathrm{C}}y \, {}^{\mathrm{I}}\mathbf{n} \left({}^{\mathrm{C}}x \right) . \tag{2.7} $$

The Frenet frame For the expression of lane keeping constraints in Chapter 4.1, we need a local Cartesian coordinate system (CS), in addition to the curvilinear CS defined above. The tangent and normal vectors of a smooth curve form a local orthonormal basis called the Frenet frame [77, Ch. 2.4]. Denoted by the superscript $^{\mathrm{F}}$, we depict this CS in Figure 2.1. The origin is the point of the curve at arc length l, ${}^{\mathrm{I}}\mathbf{r}(l)$, whereas

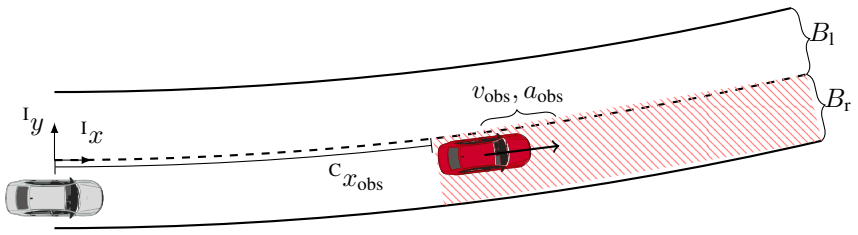


Figure 2.2: Geometry and states of the scenario model: host vehicle (gray), obstacle (red), and restricted area (dashed)

the basis vectors are given by the Frenet equations, i.e.,

$$\begin{aligned} {}^\mathrm{I}\mathbf{t}(l) &= \frac{{}^\mathrm{I}\mathbf{r}'(l)}{\|{}^\mathrm{I}\mathbf{r}'(l)\|}, \quad {}^\mathrm{I}\mathbf{r}'(l) = \frac{d\,{}^\mathrm{I}\mathbf{r}(l)}{dl} \\ {}^\mathrm{I}\mathbf{n}(l) &= \frac{{}^\mathrm{I}\mathbf{t}'(l)}{\|{}^\mathrm{I}\mathbf{t}'(l)\|}. \end{aligned} \tag{2.8}$$

The normal vector is perpendicular to the tangential vector, i.e.,

$${}^\mathrm{I}\mathbf{n}(l) = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix} {}^\mathrm{I}\mathbf{t}(l). \tag{2.9}$$

We can write the Frenet frame of a clothoid as a function of the clothoids tangent angle ${}^\mathrm{I}\tau(l)$, i.e.,

$${}^\mathrm{I}\mathbf{n}(l) = \begin{bmatrix} \cos({}^\mathrm{I}\tau(l)) \\ \sin({}^\mathrm{I}\tau(l)) \end{bmatrix}, \quad {}^\mathrm{I}\mathbf{t}(l) = \begin{bmatrix} -\sin({}^\mathrm{I}\tau(l)) \\ \cos({}^\mathrm{I}\tau(l)) \end{bmatrix}. \tag{2.10}$$

2.2 Collision Scene Model

We define a collision scene, illustrated in Figure 2.2, as the entirety of

- state vectors $\mathbf{s}(t)$ and $\mathbf{s}_\mathrm{obs}(t)$ describing, respectively, the state of the host vehicle and the obstacle at the time t,
- the lane geometry, given by a curvilinear path $\mathbf{r}(l)$ with arc length l and lane widths $B_\mathrm{r},\ B_\mathrm{l}$ that represents the lane boundaries that constrain the movement of the host vehicle.

The state vector $\mathbf{s}$ of the host vehicle is given by the states of the nonlinear single-track model as defined in Chapter 2.3. For the definition of the obstacle state vector, we make the simplifying assumption that it follows the lane with a constant longitudinal acceleration. The obstacle state vector writes as

$${}^\mathrm{C}\mathbf{s}_\mathrm{obs} = \begin{bmatrix} {}^\mathrm{C}x_\mathrm{obs} & {}^\mathrm{C}y_\mathrm{obs} & v_\mathrm{obs} & a_\mathrm{obs} \end{bmatrix}^\mathrm{T}, \tag{2.11}$$

where the ${}^{\mathrm{C}}x_{\mathrm{obs}}, {}^{\mathrm{C}}y_{\mathrm{obs}}$ are the curvilinear coordinates defined in Section 2.1, v_{obs} is the scalar velocity of the obstacle along the lane center, and a_{obs} is its acceleration.

For a compact description of the collision scene, we collect these values in a *scene tuple* $\mathcal{S}$, which provides unique and sufficient information for the criticality labeling and prediction techniques described throughout this work:

$$\mathcal{S} = (\mathbf{s},\ \mathbf{s}_{\mathrm{obs}},\ c_0,\ \kappa,\ B_{\mathrm{l}},\ B_{\mathrm{r}})\,, \tag{2.12}$$

where $\mathbf{s}_{\mathrm{obs}}$ is given in (2.11), $B_{\mathrm{l}},\ B_{\mathrm{r}}$ are the lane widths to the left and right of the lane center, c_0 is the initial curvature of the clothoid, and κ is the curvature change rate.

Obstacle prediction and restricted area For computing trajectories that avoid a rear-end collision, we further have to define a *restricted area*, i.e., an area $\mathcal{B}_{\mathrm{obs}} \subset \mathbb{R}^2$ that must not overlap with the host vehicle body. For simplicity, we assume that the host vehicle body must not enter the entire lane occupied by the obstacle. In Figure 2.2, this area is marked by the shaded red texture. We approximate the restricted area as a polytope, as detailed in Chapter 4.

We predict the restricted area of the obstacle vehicle with a prediction horizon of about two seconds using a simple prediction model. To this end, we assume that the obstacle vehicle stays on a trajectory that is parallel to the lane center (c.f., [78]) and to retain a constant longitudinal acceleration, i.e.,

$$\dot{y_{\mathrm{obs}}}(t) = 0, \quad \dot{a_{\mathrm{obs}}}(t) = 0. \tag{2.13}$$

We must also consider that vehicle or bicycle cannot immediately go backward after coming to a halt. If the obstacle decelerates, it comes to a halt at the stopping time

$$\bar{t} = -\frac{v_{\mathrm{obs},0}}{a_{\mathrm{obs},0}}, \tag{2.14}$$

where $v_{\mathrm{obs},0},\ a_{\mathrm{obs},0}$ are the initial obstacle velocity and acceleration. Consequently, the longitudinal position of the rear end of the obstacle over time is given by

$${}^{\mathrm{C}}x_{\mathrm{obs}}(t) = \begin{cases} {}^{\mathrm{C}}x_{\mathrm{obs},0} + t v_{\mathrm{obs},0} + 0.5 t^2 a_{\mathrm{obs},0}, & \text{if } t \le \bar{t} \\ {}^{\mathrm{C}}x_{\mathrm{obs}}(\bar{t}) & \text{if } t > \bar{t} \end{cases}. \tag{2.15}$$

This rather simple description of the dynamics of the obstacle which is due to the inability - at least in the absence of reliable and fast car-to-car communication - to directly measure the dynamic states of an obstacle using environment sensors like radar, lidar, and video. For longer prediction horizons of more than a few seconds, the assumption of a constant lane following by the obstacle would not be adequate. Instead, a set of maneuver hypotheses would have to be evaluated, e.g., the lane change maneuver by the obstacle.

2.3 Nonlinear Single-Track Vehicle Model

We use a nonlinear single-track model to represent the dynamics of the host vehicle during a collision avoidance maneuver. This model is based on the following simplifications [79, Ch. 10], [80, Ch. 10],

- the vehicle geometry is projected onto the two-dimensional plane,
- the front and rear tire pairs are represented by, respectively, a single tire along the longitudinal axis of the vehicle,
- the vehicle mass is a point-mass at the center-of-gravity (COG),
- roll and pitch dynamics are neglected.

The state variables represented by the model as depicted in Figure 2.3 are

- the Cartesian coordinates $x,\ y$ of the center-of-gravity,
- the norm v of the velocity of the COG,
- the body slip angle β between the longitudinal axis of the vehicle and the velocity vector of the COG,
- the yaw angle ψ of the longitudinal axis of the vehicle geometry,
- the yaw rate ω,
- the sum of the tangential tire forces $F_\mathrm{t} = F_{\mathrm{t,f}} + F_{\mathrm{t,r}}$ along the roll direction of the tires,
- the steering angle δ.

These states define the state vector of the host vehicle in a road-fixed inertial coordinate system $^\mathrm{I}$, i.e.,

$$\mathbf{s} = \begin{bmatrix} {}^\mathrm{I}x & {}^\mathrm{I}y & v & \beta & {}^\mathrm{I}\psi & \omega & \delta & F_\mathrm{t} \end{bmatrix}^\mathrm{T}. \tag{2.16}$$

The vehicle is controlled by the rates of change of the total tangential tire F_t force and steering angle. Thus, the control vector is

$$\mathbf{u} = [u_1,\ u_2], \quad u_1 = \dot{F}_\mathrm{t}, \quad u_2 = \dot{\delta}. \tag{2.17}$$

Using these state and control variables, we can take into account the body slip dynamics of the host vehicle during braking and steering maneuvers. We consider limitations on the speed of steering and braking by constraining the rates of change and magnitudes of the tire force and steering angle.

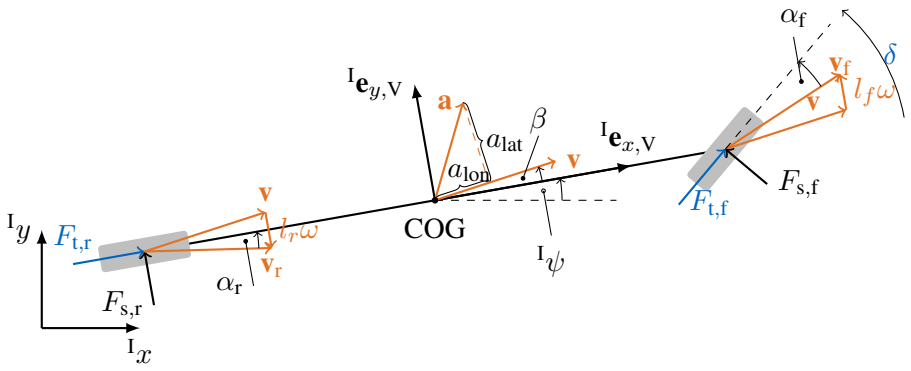


Figure 2.3: Nonlinear single-track vehicle model

Figure 2.3 illustrates the geometry and forces necessary to derive an ordinary differential equation (ODE) of the vehicle dynamics. The coordinates ${}^\mathrm{I}x, {}^\mathrm{I}y$ give the position of the COG of the host vehicle in a road-attached coordinate system ${}^\mathrm{I}$. We also define a coordinate system ${}^\mathrm{V}$ that is located at the COG, with one unit vector ${}^\mathrm{I}\mathbf{e}_{x,\mathrm{V}}$ pointing along the longitudinal axis towards the front of the vehicle. Perpendicular to ${}^\mathrm{I}\mathbf{e}_{x,\mathrm{V}}$, the second unit vector, ${}^\mathrm{I}\mathbf{e}_{y,\mathrm{V}}$, points from the COG to the left of the host vehicle. The yaw-angle ${}^\mathrm{I}\psi$ is the angle of ${}^\mathrm{I}\mathbf{e}_{x,\mathrm{V}}$ in the road-fixed CS.

During turning maneuvers, the longitudinal axis ${}^\mathrm{I}\mathbf{e}_{x,\mathrm{V}}$ of the vehicle is not aligned with the velocity vector of the COG, ${}^\mathrm{I}\mathbf{v}$. The angle from ${}^\mathrm{I}\mathbf{e}_{x,\mathrm{V}}$ to ${}^\mathrm{I}\mathbf{v}$ is called the body slip angle β, while the angle of the velocity vector is the heading angle,

$$ {}^\mathrm{I}\theta = {}^\mathrm{I}\psi + \beta. \tag{2.18} $$

Denoting the absolute velocity over ground as v, we can now write the velocity vector of the COG as

$$ {}^\mathrm{I}\mathbf{v} = \begin{bmatrix} v\cos\left({}^\mathrm{I}\psi + \beta\right) \\ v\sin\left({}^\mathrm{I}\psi + \beta\right) \end{bmatrix}. \tag{2.19} $$

Longitudinal and lateral acceleration The dynamics of the host vehicle are determined by the longitudinal and lateral acceleration of the COG as well as the angular acceleration $\dot{\omega}$. Neglecting air drag, we only have to consider the tire forces to compute the COG acceleration. In the following, the front and rear tires will be indexed by f, r. Additionally, forces in the direction of rotation of the wheel have an index t while side forces perpendicular to the turning direction of the wheel are indexed by s. To compute the acceleration of the COG, we use Newton's second law of motion, i.e.,

$$ m\,{}^\mathrm{V}\mathbf{a} = {}^\mathrm{V}\mathbf{F}_\mathrm{f} + {}^\mathrm{V}\mathbf{F}_\mathrm{r}, \tag{2.20} $$

where m is the mass of the vehicle and ${}^{\mathrm{V}}\mathbf{F}_{\mathrm{f}}$, ${}^{\mathrm{V}}\mathbf{F}_{\mathrm{r}}$ are the tire force vectors expressed in the vehicle coordinate system[1]. The front tire is rotated by the steering angle δ relative to the longitudinal vehicle axis, which is expressed by applying a rotation matrix $\mathbf{R}_\delta$ to the front tire force vector, i.e.,

$$ {}^{\mathrm{V}}\mathbf{F}_{\mathrm{f}} = \mathbf{R}_\delta \begin{bmatrix} F_{\mathrm{t,f}} \\ F_{\mathrm{s,f}} \end{bmatrix} = \begin{bmatrix} \cos(\delta) & -\sin(\delta) \\ \sin(\delta) & \cos(\delta) \end{bmatrix} \begin{bmatrix} F_{\mathrm{t,f}} \\ F_{\mathrm{s,f}} \end{bmatrix}. \tag{2.21} $$

The rear tire is aligned with the longitudinal vehicle axis, so the tire forces write as

$$ {}^{\mathrm{V}}\mathbf{F}_{\mathrm{r}} = \begin{bmatrix} F_{\mathrm{t,r}} \\ F_{\mathrm{s,r}} \end{bmatrix}. \tag{2.22} $$

By substituting the tire forces ${}^{\mathrm{V}}\mathbf{F}_{\mathrm{f}}$ and ${}^{\mathrm{V}}\mathbf{F}_{\mathrm{r}}$ into (2.20), we obtain the acceleration vector of the COG as

$$ {}^{\mathrm{V}}\mathbf{a} = \frac{1}{m}\left({}^{\mathrm{V}}\mathbf{F}_{\mathrm{r}} + {}^{\mathrm{V}}\mathbf{F}_{\mathrm{f}}\right) = \frac{1}{m}\begin{bmatrix} F_{\mathrm{t,r}} + \cos(\delta)F_{\mathrm{t,f}} - \sin(\delta)F_{\mathrm{s,f}} \\ F_{\mathrm{s,r}} + \sin(\delta)F_{\mathrm{t,f}} + \cos(\delta)F_{\mathrm{s,f}} \end{bmatrix}. \tag{2.23} $$

Next, we need to determine the acceleration component a_{lon} in the direction of the velocity vector $\mathbf{v}$ and the lateral acceleration a_{lat} that is perpendicular to $\mathbf{v}$. These accelerations are needed to compute the rate of change of the absolute velocity $\dot{v} = a_{\mathrm{lon}}$ and the yaw rate ω. To compute a_{lon} and a_{lat}, we need to project the acceleration vector ${}^{\mathrm{V}}\mathbf{a}$ onto the velocity vector. The velocity vector of the COG is rotated by the body slip angle β with respect to the longitudinal axis of the vehicle, i.e.,

$$ {}^{\mathrm{V}}\mathbf{v} = v \begin{bmatrix} \cos(\beta) \\ \sin(\beta) \end{bmatrix}. \tag{2.24} $$

The longitudinal and lateral accelerations can now be computed by projecting onto the direction of ${}^{\mathrm{V}}\mathbf{v}$, i.e.,

$$ \begin{aligned} a_{\mathrm{lon}} &= \frac{1}{v}\,{}^{\mathrm{V}}\mathbf{v}^{\mathrm{T}}\,{}^{\mathrm{V}}\mathbf{a} \\ &= \frac{1}{m}\left[\cos(\beta)\left(F_{\mathrm{t,r}} + \cos(\delta)F_{\mathrm{t,f}} - \sin(\delta)F_{\mathrm{s,f}}\right)\right. \\ &\qquad \left. + \sin(\beta)\left(F_{\mathrm{s,r}} + \sin(\delta)F_{\mathrm{t,f}} + \cos(\delta)F_{\mathrm{s,f}}\right)\right] \\ &= \frac{1}{m}\left[F_{\mathrm{t,r}}\cos(\beta) + F_{\mathrm{s,r}}\sin(\beta)\right. \\ &\qquad + F_{\mathrm{t,f}}(\sin(\beta)\sin(\delta) + \cos(\beta)\cos(\delta)) \\ &\qquad \left. + F_{\mathrm{s,f}}(\sin(\beta)\cos(\delta) - \cos(\beta)\sin(\delta))\right] \end{aligned} \tag{2.25} $$

[1] The coordinate system (CS) ${}^{\mathrm{V}}$ is an inertial CS with the origin at the vehicle COG and units vectors according to Figure 2.3. Since the CS is attached to the road, the acceleration vector does not vanish.

and by the orthogonal complement

$$\begin{aligned} a_{\text{lat}} &= (\mathbf{R}_{0.5\pi}\frac{1}{v}{}^{\text{V}}\mathbf{v})^{\text{T}}\,{}^{\text{V}}\mathbf{a} = \left(\begin{bmatrix}0 & -1\\ 1 & 0\end{bmatrix}\begin{bmatrix}\cos(\beta)\\ \sin(\beta)\end{bmatrix}\right)^{\text{T}}{}^{\text{V}}\mathbf{a} \\ &= \frac{1}{m}\left[-\sin(\beta)\left(F_{\text{t,r}}+\cos(\delta)F_{\text{t,f}}-\sin(\delta)F_{\text{s,f}}\right)\right. \\ &\qquad \left.+\cos(\beta)\left(F_{\text{s,r}}+\sin(\delta)F_{\text{t,f}}+\cos(\delta)F_{\text{s,f}}\right)\right] \\ &= \frac{1}{m}\left[-\sin(\beta)F_{\text{t,r}}+F_{\text{s,r}}\cos(\beta)\right. \\ &\qquad +F_{\text{t,f}}(-\sin(\beta)\cos(\delta)+\cos(\beta)\sin(\delta)) \\ &\qquad \left.+F_{\text{s,f}}(\sin(\beta)\sin(\delta)+\cos(\beta)\cos(\delta))\right]. \end{aligned} \tag{2.26}$$

Using angle sum identities[2], these expressions can be simplified to

$$\begin{aligned} a_{\text{lon}} &= \frac{1}{m}\left[F_{\text{t,r}}\cos(\beta)+F_{\text{s,r}}\sin(\beta)+F_{\text{t,f}}\cos(\beta-\delta)+F_{\text{s,f}}\sin(\beta-\delta)\right], \\ a_{\text{lat}} &= \frac{1}{m}\left[F_{\text{s,r}}\cos(\beta)-F_{\text{t,r}}\sin(\beta)-F_{\text{t,f}}\sin(\beta-\delta)+F_{\text{s,f}}\cos(\beta-\delta)\right]. \end{aligned} \tag{2.27}$$

Assuming that the vehicle does not drive backward, i.e., $v > 0\ \forall t$, we write the rate of change of the velocity over ground as

$$\dot{v} = a_{\text{lon}}. \tag{2.28}$$

Given the lateral acceleration a_{lat} and the velocity over ground, we can compute the yaw rate of the vehicle. Since a_{lat} is equal to the centripetal acceleration of the COG, we can relate it to the rate of change of the heading angle, i.e.,

$$a_{\text{lat}} = v\dot{\theta} = v(\omega+\dot{\beta}). \tag{2.29}$$

We can now write the rate of change of the body slip angle as

$$\dot{\beta} = \frac{a_{\text{lat}}}{v} - \omega. \tag{2.30}$$

At this point, we have derived the differential equations for the position, the velocity, and the body slip angle.

Angular Acceleration The angular momentum of the vehicle around its center-of-gravity is related to the yaw rate ω by

$$\mathbf{L} = I\omega = I\begin{bmatrix}0 & 0 & \omega\end{bmatrix}^{\text{T}}, \tag{2.31}$$

[2] $\sin(\beta)\sin(\delta)+\cos(\beta)\cos(\delta)=\cos(\beta-\delta),\ \sin(\beta)\cos(\delta)-\cos(\beta)\sin(\delta)=\sin(\beta-\delta)$ [81]

where I is the so-called moment of inertia. The rate of change of the angular momentum is proportional to the sum of torques $\mathbf{q}_{\mathrm{f}},\ \mathbf{q}_{\mathrm{r}} \in \mathbb{R}^3 \in$, i.e.,

$$\dot{\mathbf{L}} = I\dot{\omega} = \mathbf{q}_{\mathrm{f}} + \mathbf{q}_{\mathrm{r}}. \tag{2.32}$$

Let ${}^{\mathrm{V}'}$ denote the three-dimensional inertial coordinate system of the vehicle defined by the basis $\mathbf{e}_{x,\mathrm{V}}$, $\mathbf{e}_{y,\mathrm{V}}$, and $\mathbf{e}_{z,\mathrm{V}} = \mathbf{e}_{x,\mathrm{V}} \times \mathbf{e}_{y,\mathrm{V}}$ (ref., Figure 2.3). Then, the torques acting on the COG write as

$$\mathbf{q}_{\mathrm{f}} = {}^{\mathrm{V}'}\mathbf{r}_{\mathrm{f}} \times {}^{\mathrm{V}'}\mathbf{F}_{\mathrm{f}}, \quad \mathbf{q}_{\mathrm{r}} = {}^{\mathrm{V}'}\mathbf{r}_{\mathrm{r}} \times {}^{\mathrm{V}'}\mathbf{F}_{\mathrm{r}}, \tag{2.33}$$

where $\times$ is the cross-product and ${}^{\mathrm{V}'}\mathbf{r}_{\mathrm{f}}$, ${}^{\mathrm{V}'}\mathbf{r}_{\mathrm{r}}$ denote the positions where the forces act on the vehicle body, relative to the COG. Denoting the distances of the wheel centers from the COG as l_f and l_r, the wheel positions in the vehicle CS are given by

$${}^{\mathrm{V}'}\mathbf{r}_{\mathrm{f}} = \begin{bmatrix} l_f & 0 & 0 \end{bmatrix}^{\mathrm{T}}, \quad {}^{\mathrm{V}'}\mathbf{r}_{\mathrm{r}} = \begin{bmatrix} -l_r & 0 & 0 \end{bmatrix}^{\mathrm{T}}. \tag{2.34}$$

The resulting torques are

$$\mathbf{q}_{\mathrm{f}} = \begin{bmatrix} l_f \\ 0 \\ 0 \end{bmatrix} \times \begin{bmatrix} \cos(\delta)F_{\mathrm{t,f}} - \sin(\delta)F_{\mathrm{s,f}} \\ \sin(\delta)F_{\mathrm{t,f}} + \cos(\delta)F_{\mathrm{s,f}} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ l_f \sin(\delta)F_{\mathrm{t,f}} + l_f \cos(\delta)F_{\mathrm{s,f}} \end{bmatrix}, \tag{2.35}$$

$$\mathbf{q}_{\mathrm{r}} = \begin{bmatrix} -l_r \\ 0 \\ 0 \end{bmatrix} \times \begin{bmatrix} F_{\mathrm{t,r}} \\ F_{\mathrm{s,r}} \\ 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ -l_r F_{\mathrm{s,r}} \end{bmatrix}. \tag{2.36}$$

By inserting the torques of the front and rear tires into (2.32), we can compute the yaw acceleration, i.e.,

$$I\dot{\omega} = I \begin{bmatrix} 0 \\ 0 \\ \dot{\omega} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ l_f \sin(\delta)F_{\mathrm{t,f}} + l_f \cos(\delta)F_{\mathrm{s,f}} \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ -l_r F_{\mathrm{s,r}} \end{bmatrix},$$

$$\dot{\omega} = \frac{1}{I}\left(l_f \sin(\delta)F_{\mathrm{t,f}} + l_f \cos(\delta)F_{\mathrm{s,f}} - l_r F_{\mathrm{s,r}}\right). \tag{2.37}$$

The derived dynamic equations all depend on the contact forces between the front and rear tire of the single-track model. As in [48], we assume that the total tangential tire force F_{t}, introduced at the beginning of this chapter, is distributed proportionally to the front and rear wheel, i.e.,

$$F_{\mathrm{t,f}} = k_{\mathrm{b}} F_{\mathrm{t}}, \quad F_{\mathrm{t,r}} = (1 - k_{\mathrm{b}})F_{\mathrm{t}}, \tag{2.38}$$

with the proportional braking factor k_{b} given as a model parameter. The tire side forces $F_{\mathrm{s,f}},\ F_{\mathrm{s,r}}$ depend on the choice of a tire model, which is detailed in Section 2.4.

Differential equations By collecting (2.17), (2.19), (2.19), (2.30), (2.37), we are now able to give an ordinary differential equation for the trajectory of the nonlinear single-track model,

$$\begin{bmatrix} {}^{\mathrm{I}}\dot{x} \\ {}^{\mathrm{I}}\dot{y} \\ \dot{v} \\ \dot{\beta} \\ \dot{\psi} \\ \dot{\omega} \\ \dot{F}_{\mathrm{t}} \\ \dot{\delta} \end{bmatrix} = \begin{bmatrix} v\cos\left({}^{\mathrm{I}}\psi + \beta\right) \\ v\sin\left({}^{\mathrm{I}}\psi + \beta\right) \\ a_{\mathrm{lon}} \\ a_{\mathrm{lat}}/v - \omega \\ \omega \\ \frac{1}{I}(F_{\mathrm{s,f}}\cos(\delta)l_f + F_{\mathrm{t,f}}\sin(\delta)l_f - F_{\mathrm{s,r}}l_r) \\ u_1 \\ u_2 \end{bmatrix}. \tag{2.39}$$

In the remainder, this ODE is abbreviated as

$$\dot{\mathbf{s}} = \mathrm{f}(\mathbf{s},\ \mathbf{u}). \tag{2.40}$$

This nonlinear single-track model has previously been used in [46, 49, 48] in the context of optimal control. There, the control input is the steering angle and tangential tire force instead of their derivatives. It would be worthwhile to use a two-track model that considers roll and pitch dynamics for the simulations due to its even better fidelity (c.f., [79, Ch. 10] and [80]). Most importantly, the roll and pitch dynamics influence the distribution of forces among the tires of the vehicle and thus change the control variables necessary for performing a particular evasion trajectory. Taking roll and pitch into account would require a larger number of dynamic states that would have to be simulated and optimized, as well as more model parameters.

2.4 Tire Models

We consider linear and nonlinear tire models for the tire side forces $F_{\mathrm{s,f}},\ F_{\mathrm{s,r}}$ as well as the coupling of side and tangential forces. The tire models are illustrated in Figure 2.4 and explained in the following.

Side slip angle All tire models depend on the so-called side slip angle of the tire. The side slip angle is defined as the angle between the roll direction of the tire and the velocity vector of the tire over ground as illustrated in Figure 2.3. The velocity of the tire has a rotation component due to the yaw rate of the vehicle as well as a translatory component due to the velocity over ground of the COG, i.e.,

$$\begin{aligned} {}^{\mathrm{V}}\mathbf{v}_{\mathrm{f}} &= {}^{\mathrm{V}}\mathbf{v} + {}^{\mathrm{V}}\mathbf{v}_{\mathrm{rot,f}} \\ {}^{\mathrm{V}}\mathbf{v}_{\mathrm{r}} &= {}^{\mathrm{V}}\mathbf{v} + {}^{\mathrm{V}}\mathbf{v}_{\mathrm{rot,r}} \end{aligned} \tag{2.41}$$

By inspection, the rotational components are given by

$$ {}^{\mathrm{V}}\mathbf{v}_{\mathrm{rot,f}} = \begin{bmatrix} 0 & l_f\omega \end{bmatrix}^{\mathrm{T}}, \quad {}^{\mathrm{V}}\mathbf{v}_{\mathrm{rot,r}} = \begin{bmatrix} 0 & -l_r\omega \end{bmatrix}^{\mathrm{T}}, \tag{2.42} $$

whereas the translatory component is the COG velocity in the vehicle CS which is common to the front and rear tires, i.e.,

$$ {}^{\mathrm{V}}\mathbf{v} = \begin{bmatrix} v\cos(\beta) & v\sin(\beta) \end{bmatrix}^{\mathrm{T}}. \tag{2.43} $$

The tire velocities become

$$ {}^{\mathrm{V}}\mathbf{v}_{\mathrm{f}} = \begin{bmatrix} v\cos(\beta) \\ v\sin(\beta) + l_f\omega \end{bmatrix}, \quad {}^{\mathrm{V}}\mathbf{v}_{\mathrm{r}} = \begin{bmatrix} v\cos(\beta) \\ v\sin(\beta) - l_r\omega \end{bmatrix}. \tag{2.44} $$

To obtain the tire slip angle α, we need to compute the angle of the tire velocity vectors, i.e.,

$$ \begin{aligned} {}^{\mathrm{V}}\angle\mathbf{v}_{\mathrm{f}} &= \arctan\left(\frac{v\sin(\beta) + l_f\omega}{v\cos(\beta)}\right), \\ {}^{\mathrm{V}}\angle\mathbf{v}_{\mathrm{f}} &= \arctan\left(\frac{v\sin(\beta) - l_r\omega}{v\cos(\beta)}\right), \end{aligned} \tag{2.45} $$

where $\angle\mathbf{v}$ maps the vector $\mathbf{v}$ to its polar angle. The tire slip angles are given by the differences of the steering angles of the front and rear tire and the tire velocity angles, i.e.,

$$ \begin{aligned} \alpha_{\mathrm{f}} &= \delta - \arctan\left(\frac{l_f\omega + v\sin(\beta)}{v\cos(\beta)}\right), \\ \alpha_{\mathrm{r}} &= 0 - \arctan\left(\frac{v\sin(\beta) - l_r\omega}{v\cos(\beta)}\right) = \arctan\left(\frac{l_r\omega - v\sin(\beta)}{v\cos(\beta)}\right). \end{aligned} \tag{2.46} $$

Linear Model and Magic Tire Formula The simplest model for the tire side force is a linear function of the tire side slip angle, i.e.,

$$ F_{\mathrm{s,f}} = k_{\mathrm{f}}\alpha_{\mathrm{f}}, \quad F_{\mathrm{s,r}} = k_{\mathrm{r}}\alpha_{\mathrm{r}}, \tag{2.47} $$

where the proportionality constants $k_{\mathrm{f}},\ k_{\mathrm{r}}$ are referred to as the cornering stiffnesses of the tires. For simplicity, we use the same cornering stiffness for the front and rear tires, $k_{\mathrm{f}} = k_{\mathrm{r}} = k$.

Compared to more complex tire models, an essential deficiency of the linear tire model is that it does not limit the magnitude of the tire forces. For real tires, the side forces are limited by the fact that the tire starts to slide on the road surface as the side slip angle α increases, depending on the tire and load.

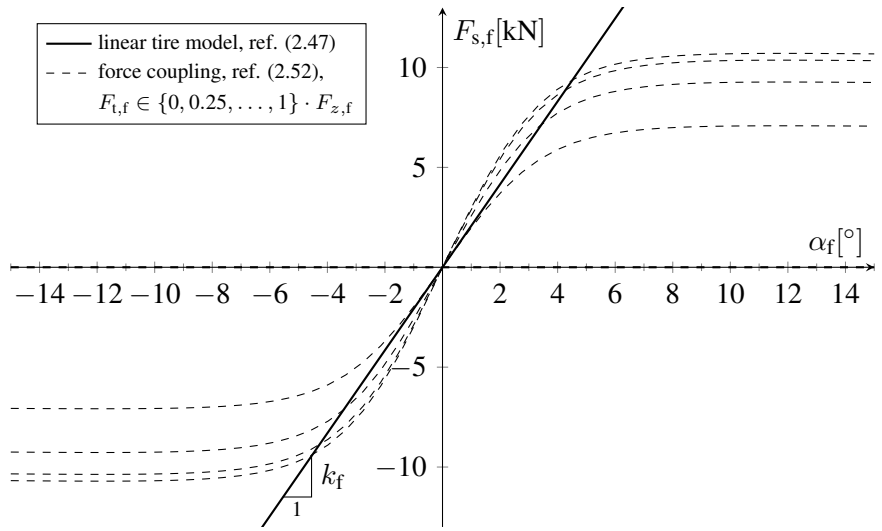


Figure 2.4: Linear and nonlinear models for the tire side force $F_{\mathrm{s,f}}$ for varying tangential tire forces $F_{\mathrm{t,f}}$. Vertical tire force $F_{z,\mathrm{f}} = \frac{l_r}{l_f+l_r} mg$. Cornering stiffness for the linear model $k_{\mathrm{f}} = 2BCD$. Parameters are given in Tables 2.1 and 4.2.

The so-called *magic tire formula* [82] is an empirical model (as opposed to a physics-based model) designed to fit the side forces measured under various experimental conditions. The general shape of the magic tire formula is depicted in Figure 2.4. It models the saturation of the side force as a function of the side slip angle and is given by

$$F_{\mathrm{s}}(\alpha) = D \sin\left(C \arctan(B\Phi)\right), \tag{2.48}$$
$$\Phi = (1-E)\alpha + \left(\frac{E}{B}\right) \arctan(B\alpha).$$

Due to its simplicity and its low computational cost, it is widely used in the active safety domain.

The cornering stiffness of the tire $k_{\mathrm{f,r}}$ corresponds to the derivative of the side force around $\alpha = 0$ and is given by [82] as $k'_{\mathrm{f,r}} = BCD$. Since we use a single-track model, each tire represents the two tires of the front or rear axle of the vehicle, respectively. Thus, it has twice the vertical load and exerts twice the side force as a front or rear tire in a model with four wheels. For the cornering stiffness in the linear tire model, we account for this observation by doubling the cornering stiffness, i.e.,

$$k_{\mathrm{f,r}} = 2BCD. \tag{2.49}$$

The tire parameters used throughout this work are listed in Table 2.1 and are derived from [82, Tbl. 1]. These parameters depend on the vertical tire forces

Table 2.1: Parameters used for the magic tire formula (2.48)

parameter	symbol	value	unit
stiffness factor	B	0.239	$1/^\circ$
shape factor	C	1.19	1
peak factor	D	3750	N
curvature factor	E	-0.678	1

$F_{z,\mathrm{f}},\ F_{z,\mathrm{r}}$ and, thus, are affected by load transfer between the front and rear axles. Choosing constant tire parameters can therefore only be considered an approximation of the tire behavior during evasion maneuvers.

Force Ellipse Next, we consider the limitation of the norm of the tire force vector due to the limited tire-to-surface friction [82, 83]. To this end, we adopt the force coupling model used in [48]. The norm of the tire force vector is limited by

$$\sqrt{F_{\mathrm{s}}^2 + F_{\mathrm{t}}^2} \leq \mu F_z, \tag{2.50}$$

where μ is the friction coefficient between the tire and the road surface and F_z is the vertical tire force. Typical limits for the friction coefficient μ are up to 0.8 to 0.9 for dry asphalt and in the range of 0.5 to 0.7 for wet asphalt [84, Tbl. 1]. To satisfy this inequality, the tangential tire force has to be constrained to

$$-\mu F_z \leq F_{\mathrm{t}} \leq \mu F_z. \tag{2.51}$$

The tire side force satisfying the force ellipse constraint can now be computed as [48],

$$F_{\mathrm{s}}(\alpha, F_{\mathrm{t,f}}) = f_{\mathrm{s}} \sqrt{1 - \left(\frac{F_{\mathrm{t}}}{\mu F_z}\right)^2}, \tag{2.52}$$

where f_{s} is the uncoupled side force. It is a modified version of the magic formula where the peak parameter D of the magic formula (2.48) is replaced by the friction coefficient and vertical load, i.e.,

$$\begin{aligned} f_{\mathrm{s}}(\alpha) &= \mu F_z \sin\left(C \arctan(B\Phi)\right), \\ \Phi &= (1-E)\alpha + \left(\frac{E}{B}\right) \arctan(B\alpha). \end{aligned} \tag{2.53}$$

We can check that the norm of the force vector never exceeds μF_z, i.e.,

$$\begin{aligned} F_{\mathrm{s}}^2 + F_{\mathrm{t}}^2 &= f_{\mathrm{s}}^2(\alpha)\left(1 - \left(\frac{F_{\mathrm{t}}}{\mu F_z}\right)^2\right) + F_{\mathrm{t}}^2 \\ &\leq (\mu F_z)^2\left(1 - \left(\frac{F_{\mathrm{t}}}{\mu F_z}\right)^2\right) + F_{\mathrm{t}}^2, \\ F_{\mathrm{s}}^2 + F_{\mathrm{t}}^2 &\leq (\mu F_z)^2 - F_{\mathrm{t}}^2 + F_{\mathrm{t}}^2 = (\mu F_z)^2. \end{aligned} \tag{2.54}$$

It remains to compute the load on the front and rear tire of the single-track model, i.e.,

$$F_{z,\mathrm{f}} = mg\frac{l_r}{l_f + l_r}, \quad F_{z,\mathrm{r}} = mg\frac{l_f}{l_f + l_r}, \tag{2.55}$$

where m is the lumped mass at the center of gravity of the vehicle. Figure 2.4 depicts the coupled tire model of the front tire of the single-track model for varying tangential tire forces and side slip angles.

Estimation of the Friction Coefficient For active safety systems and autonomous driving, the friction coefficient of the surface in front of the vehicle must be estimated and predicted. Model-based parameter estimation techniques like the Kalman filter are prominent approaches for the estimation of the friction coefficient [85, 86]. Other approaches include the use of near-infrared spectroscopy [87] and the analysis of the tire acoustics [88]. In [89], the recognition of luminance patterns of a front-facing mono video camera is combined with analyzing peaks of the acoustic spectrum of the tire for varying vehicle speeds and road surface characteristics like dry, wet, and snowy. The reader is referred to [90] and [91] for a survey of tire modeling and online parameter estimation.

Chapter 3

Numerical Optimal Control using Direct Shooting Methods

This chapter introduces direct shooting methods for numerical optimal control with a focus on Direct Multiple Shooting [44] as used for the computation of criticality labels throughout this work. A general form of the continuous time optimal control

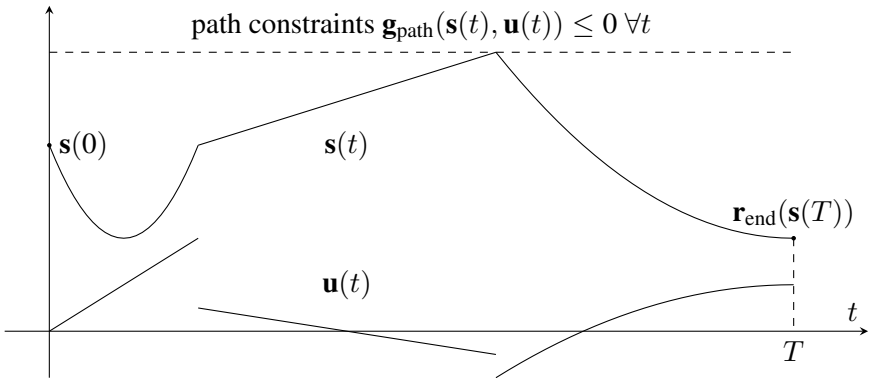


Figure 3.1: Illustration of an optimal control problem [92]

problem writes as [92],

$$\begin{aligned} \max_{\mathbf{s},\mathbf{u},T} \int_0^T \mathrm{L}(\mathbf{s}(t),\mathbf{u}(t))dt + E(\mathbf{s}(T)),\ \text{s.t.} \\ \mathbf{s}(0) - \mathbf{s}_0 &= \mathbf{0}, \\ \dot{\mathbf{s}}(t) - f(\mathbf{s}(t),\mathbf{u}(t)) &= \mathbf{0},\ \forall t \in [0,T], \\ \mathbf{g}_{\text{path}}(\mathbf{s}(t),\mathbf{u}(t)) &\leq \mathbf{0},\ \forall t \in [0,T], \\ \mathbf{r}_{\text{end}}(\mathbf{s}(T)) &= \mathbf{0}, \end{aligned} \tag{3.1}$$

which comprises a fixed initial value $\mathbf{s}_0$, the ordinary differential equation (ODE) constraint $\dot{\mathbf{s}}(t) = f(\mathbf{s}(t),\mathbf{u}(t)$ as well as path and endpoint constraints encoded by the functions $\mathbf{g}_{\text{path}}(\mathbf{s}(t),\mathbf{u}(t))$ and $\mathbf{r}_{\text{end}}(\mathbf{s}(T))$, respectively. The shorthand *s.t.* stands for *subject to*. Figure 3.1 provides an illustration of this OCP adapted from [92].

In this formulation, the length of the time horizon T is a free optimization variable in the OCP. Such variable horizon problems have many applications throughout robotics and are used, e.g., in [50] for finding minimal time evasion maneuvers for active safety systems. In the following, as throughout this work, we consider the case of a fixed horizon length.

Direct shooting methods are one way of numerically solving optimal control problems by discretizing the prediction horizon into subintervals and integrating the ODE constraint. The discretization yields a nonlinear optimization problem that is then solved using nonlinear optimization solvers. We assume that the time interval is divided into N subintervals of equal length such that

$$t_n = n\Delta t,\ \Delta t = \frac{T}{N},\ n = 0,\dots,N. \tag{3.2}$$

In direct methods of numerical optimal control, the control inputs $\mathbf{u}(t)$ are parameterized using piecewise constant functions on subintervals of the time interval,

$$\mathbf{u}(t) = \sum_{n=0}^{N-1} \mathbf{u}[n]\,\mathrm{Ind}_{[t_n,t_{n+1})}(t), \quad \mathrm{Ind}_{[t_n,t_{n+1})}(t) = \begin{cases} 1 & \text{if } t \in [t_n,t_{n+1}) \\ 0 & \text{otherwise,} \end{cases} \tag{3.3}$$

where $\mathrm{Ind}_{[t_n,t_{n+1})}(t)$ is the indicator function on the subinterval $[t_n,t_{n+1})$ and $\mathbf{u}[n]$ are the control parameters[1] .

3.1 Direct Single Shooting

In direct single shooting, the optimal control problem is solved by expressing the state over the entire time interval $[0,T]$ as the solution of an initial value problem (IVP)

[1] The piecewise constant model can represent smoother control functions on each subinterval by introducing additional state variables.

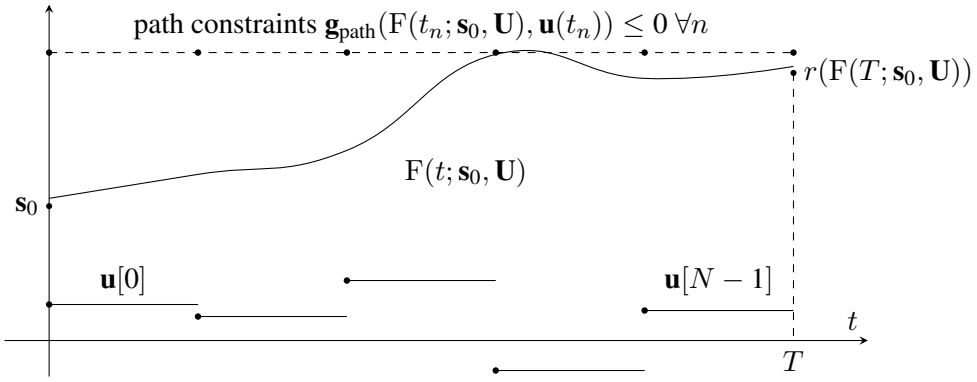


Figure 3.2: Illustration of direct single shooting [92]

starting with the initial state $\mathbf{s}_0$, i.e.,

$$\hat{\mathbf{s}}(t;\mathbf{U}) = \mathrm{F}(t;\mathbf{s}_0,\mathbf{U}), \forall t \in [0,T],\ \mathbf{U} = \mathbf{u}[0],\ldots,\mathbf{u}[N-1]. \tag{3.4}$$

The function $\mathrm{F}(t;\mathbf{s}_0,\mathbf{U})$ represents numerical integration with the initial value $\mathbf{s}_0$. The single shooting method leads to the finite dimensional optimization problem illustrated in Figure 3.2 and written as [92]

$$\begin{aligned} \min_{\mathbf{U}=\mathbf{u}[0],\ldots,\mathbf{u}[N-1]} & \int_0^T \mathrm{L}(\hat{\mathbf{s}}(t;\mathbf{U}),\mathbf{u}(t;\mathbf{U}))dt + E(\hat{\mathbf{s}}(T;\mathbf{U})),\ \text{s.t.} \\ \mathbf{g}_{\text{path}}(\hat{\mathbf{s}}(t_n;\mathbf{U}),\mathbf{u}(t_n;\mathbf{U})) &\leq \mathbf{0},\ i = 1,\ldots,N-1, \\ \mathbf{r}_{\text{end}}(\hat{\mathbf{s}}(T;\mathbf{U})) &= \mathbf{0}. \end{aligned} \tag{3.5}$$

In this problem formulation, the path constraints are enforced at the beginning of each subinterval at the discrete times $t = t_n$. This leads to a coupling over the entire time interval between the initial state $\mathbf{s}_0$, path constraints at times t_n, and the control variables $\mathbf{u}[m],\ m \leq n$. Consequently, the resulting optimization problem often turns out to be highly nonlinear and unstable [92].

3.2 Direct Multiple Shooting

In comparison to the single-shooting formulation, the multiple shooting formulation [44] also optimizes the state variables $\mathbf{s}[0],\ldots,\mathbf{s}[N]$, thus introducing a larger number of optimization variables and constraints. Instead of solving a single initial-value problem as in the single-shooting formulation, the ODE is solved on each subinterval starting at the states $\mathbf{s}[n]$ which leads to a set of $N-1$ initial value problems, i.e.,

$$\dot{\mathbf{s}}(t) = \mathrm{f}(\mathbf{s}(t),\mathbf{u}[n]), \quad \mathbf{s}(t_n) = \mathbf{s}[n],\ \forall n = 0,\ldots,N-1. \tag{3.6}$$

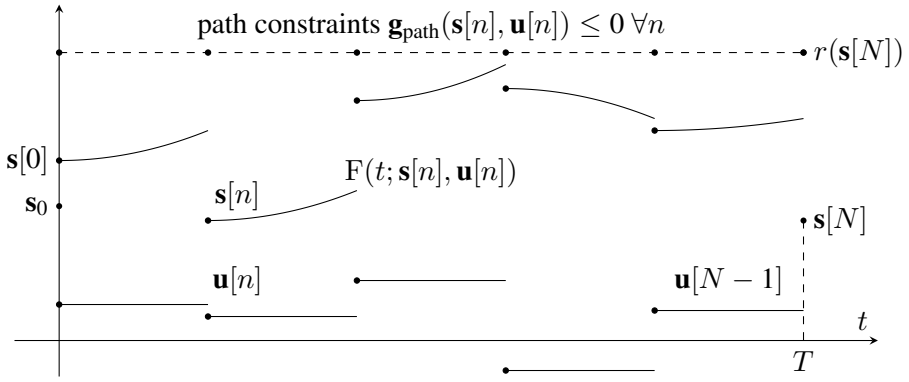


Figure 3.3: Illustration of direct multiple shooting [92]

The decoupling of the state trajectory into shooting intervals alleviates the high nonlinearity of the single-shooting formulation [92].

As illustrated in Figure 3.3, the IVPs are solved using numerical integration on each subinterval,

$$\hat{\mathbf{s}}_n(t) = \mathrm{F}(t; \mathbf{s}[n], \mathbf{u}[n]), \forall t \in [t_n, t_{n+1}],\ \forall n = 0, \ldots, N-1, \tag{3.7}$$

where $\mathrm{F}(\bullet)$ represents a suitable numerical integration method. To still obtain a continuous solution of the differential equations, a set of continuity or *shooting* constraints are introduced which enforce that the solution of the state integration at the end of each subinterval $[t_n, t_{n+1})$ matches the state variable $\mathbf{s}[n+1]$ of the next subinterval,

$$\mathbf{s}[n+1] - \mathrm{F}(\mathbf{s}[n], \mathbf{u}[n]) = \mathbf{0},\ \forall n = 0, \ldots, N-1. \tag{3.8}$$

By comparison to (3.4), we can see that direct multiple shooting solves many IVPs with a short integration interval compared to single shooting, where the ODE is integrated over the entire time horizon.

The necessary complexity of the integration routine depends primarily on what is referred to as the *stiffness* of the ODE $\mathrm{f}(\mathbf{s}, \mathbf{u})$. A stiff ODE designates an ODE whose matrix $\frac{\partial \mathrm{f}(\mathbf{s},\mathbf{u})}{\partial \mathbf{s}}$ is poorly conditioned [93, Ch. 12.1]. When solving optimal control problems, this instability can lead to slow convergence or a complete lack of convergence. The fact that the multiple shooting method integrates the ODE over many small subintervals $[t_n, t_{n+1}]$ instead of a one long integration interval $[0, T]$ alleviates the problem of the numerical integration of stiff ODEs.

The cost function of the OCP is written as the sum of the costs on each time interval, which the cost contribution of the n-th interval written as

$$l_n(\mathbf{s}[n], \mathbf{u}[n]) = \int_{t_n}^{t_{n+1}} \mathrm{L}(\mathbf{s}[n], \mathbf{u}[n]) dt. \tag{3.9}$$

The nonlinear optimization problem resulting from the multiple shooting discretization reads as,

$$\begin{aligned}
\min_{\substack{\mathbf{s}[n],n=0,\dots,N \\ \mathbf{u}[n],n=0,\dots,N-1}} \sum_{0}^{N-1} l_n(\mathbf{s}[n],\mathbf{u}[n]) + E(\mathbf{s}[N]) \text{ s.t.} \\
\mathbf{s}[0] - \mathbf{s}_0 = \mathbf{0}, \\
\mathbf{s}[n+1] - \mathrm{F}(t_{n+1};\mathbf{s}[n],\mathbf{u}[n]) = \mathbf{0},\ n = 0,\dots,N-1, \\
\mathbf{g}_{\text{path}}(\mathbf{s}[n],\mathbf{u}[n]) \leq \mathbf{0},\ n = 0,\dots,N-1, \\
\mathbf{r}_{\text{end}}(\mathbf{s}[N]) = \mathbf{0}.
\end{aligned} \tag{3.10}$$

Compared to single shooting, the multiple shooting formulation introduces a lot of additional variables in the form of the state parameters $\mathbf{s}[n]$ and additional shooting constraints. However, the nonlinearity of the optimization problem and the coupling between the optimization variables is significantly reduced compared to the single shooting formulation. This is due to the short integration intervals of the IVPs and the fact that the shooting constraints only have to be fulfilled once the optimization has converged. Additionally, using the states as optimization variables has the advantage that knowledge about the state trajectory $\mathbf{s}(t)$ can be used as an initial guess for iterative solvers.

An important algorithmic component for the solution of large NLP is the use of *automatic differentiation* for obtaining the gradient and Hessian matrices of the Lagrangian of the NLP. We use the CasADI [94] library which provides automatic differentiation coupled with a Python interface for the programmatic formulation of optimal control problems.

Another property of multiple shooting that reduces its computational complexity is the fact that the resulting optimization problem is highly sparse. This can be seen by analyzing the Lagrangian of (3.10), i.e.,

$$\begin{aligned}
\mathcal{L}(\mathbf{s},\mathbf{u},\lambda,\mathbf{z}) = &\sum_{n=0}^{N-1} l_n(\mathbf{s}[n],\mathbf{u}[n]) + E(\mathbf{s}[N]) + \lambda_{\text{start}}^{\mathrm{T}}(\mathbf{s}[0]-\mathbf{s}_0) + \lambda_{\text{end}}^{\mathrm{T}}(\mathbf{r}_{\text{end}}(\mathbf{s}[N])) \\
&+ \sum_{n=0}^{N-1} \mathbf{z}_n^{\mathrm{T}}\mathbf{g}_{\text{path}}(\mathbf{s}[n],\mathbf{u}[n]) + \sum_{n=0}^{N-1} \lambda_n^{\mathrm{T}}(\mathbf{s}[n+1] - \mathrm{F}(t_{n+1};\mathbf{s}[n],\mathbf{u}[n])).
\end{aligned} \tag{3.11}$$

Collecting the primal optimization variables as

$$\begin{aligned}
\mathbf{w}_n &= \begin{bmatrix} \mathbf{s}[n] \\ \mathbf{u}[n] \end{bmatrix} \forall n = 0,\dots,N-1,\ \mathbf{w}_N = \mathbf{s}[N], \\
\mathbf{w} &= \begin{bmatrix} \mathbf{w}_0^{\mathrm{T}} & \dots & \mathbf{w}_N^{\mathrm{T}} \end{bmatrix}^{\mathrm{T}},
\end{aligned} \tag{3.12}$$

the Lagrangian of the multiple-shooting problem can be written as the sum of terms that only depend on the subsets $\mathbf{w}_n$ of $\mathbf{w}$, i.e.,

$$\mathcal{L}(\mathbf{w}, \lambda, \mathbf{z}) = \sum_{n=0}^{N} \mathcal{L}_n(\mathbf{w}_n, \lambda, \mathbf{z}). \tag{3.13}$$

From here it is easy to deduce the block-diagonal structure of the Hessian matrix $\nabla^2_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \lambda, \mathbf{z})$ of the Lagrangian [92], i.e.,

$$\begin{aligned} \nabla_{\mathbf{w}_n} \mathcal{L}(\mathbf{w}, \lambda, \mathbf{z}) &= \nabla_{\mathbf{w}_n} \mathcal{L}_n(\mathbf{w}_n, \lambda, \mathbf{z}) \\ \nabla_{\mathbf{w}_m} \nabla_{\mathbf{w}_n} \mathcal{L}(\mathbf{w}, \lambda, \mathbf{z}) &= \mathbf{0} \; \forall m \neq n. \end{aligned} \tag{3.14}$$

Like the Hessian of the Lagrangian, the Jacobian of the equality constraints has a block-sparse structure.

To achieve a high computational efficiency, a sparsity-exploiting optimization solver must be used for finding solutions to the multiple shooting formulation. We use the sparse optimization solver IPOPT [95] which implements a primal-dual interior-point method as described in the following. Other popular solvers in this context are based on sparse sequential quadratic programming, e.g., SNOPT [96] and MUSCOD-II [97].

3.3 Primal-Dual Interior-Point Optimization

Interior-point methods are an attractive choice for solving inequality constrained nonlinear optimization problems if there is a large number of inequality constraints as it is often the case for nonlinear optimal control problems. Following the expositions in [95] and [98, Ch. 11], we introduce the algorithmic basis of the primal-dual interior-point (PDIP) algorithm for solving nonlinear optimization problems of the type

$$\min_{\mathbf{w} \in \mathbb{R}^m} \mathrm{f}(\mathbf{w}) \text{ s.t.} \quad \mathrm{b}(\mathbf{w}) = \mathbf{0}, \quad \mathbf{w} \geq \mathbf{0}, \tag{3.15}$$

where we assume that the cost function f and equality function b are twice continuously differentiable. Problems with general nonlinear inequality constraints $\mathrm{c}(\mathbf{w}) \geq \mathbf{0}$ can be rewritten in this form by introducing slack variables [95].

The Karush-Kuhn-Tucker (KKT) optimality conditions express necessary conditions that have to be fulfilled by optimizers $\mathbf{w}$ of (3.15) [98]. The Lagrangian function of (3.15) is given by

$$\mathcal{L}(\mathbf{w}, \lambda, \mathbf{z}) = \mathrm{f}(\mathbf{w}) - \lambda^{\mathrm{T}} \mathrm{b}(\mathbf{w}) - \mathbf{z}^{\mathrm{T}} \mathbf{w}, \tag{3.16}$$

whereas the KKT conditions write as

$$\begin{aligned} \nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w},\lambda,\mathbf{z}) = \nabla_{\mathbf{w}}\,\mathrm{f}(\mathbf{w}) - \nabla_{\mathbf{w}}\lambda^{\mathrm{T}}\,\mathrm{b}(\mathbf{w}) - \mathbf{z} = \mathbf{0}, &\quad \text{Stationarity} \\ \mathrm{diag}(\mathbf{z})\mathbf{w} = \mathbf{0}, &\quad \text{Complementary Slackness} \\ \mathbf{w} \geq \mathbf{0}, \quad \mathrm{b}(\mathbf{w}) = \mathbf{0}, &\quad \text{Primal Feasibility} \\ \mathbf{z} \geq \mathbf{0}. &\quad \text{Dual Feasibility} \end{aligned} \tag{3.17}$$

For a large number of optimization variables and inequality constraints, a direct solution of the KKT conditions is typically impractical.

The barrier method Before treating the primal-dual interior-point method, it is instructive to consider the barrier approximation to (3.15). Barrier methods solve this nonlinear optimization problem by solving a sequence of equality constrained problems where the inequality constraints are included in the cost function, i.e.,

$$\min_{\mathbf{w}} \mathrm{f}(\mathbf{w}) - \mu \sum_{i=1}^{m} \ln(w_i) \,\text{s.t.}\, \mathrm{b}(\mathbf{w}) = \mathbf{0}. \tag{3.18}$$

The logarithm term of the cost function assigns a rapidly increasing cost to optimization variables that get close to the boundary of the feasible set. The optimal value of the barrier problem (3.18) is an upper bound to the optimal value of (3.15) [98]. For convex problems, it is shown in [98] that the optimizer $\mathbf{w}_\mu^*$ of (3.18) is suboptimal by $m\mu$. Since the log-barrier cost function is not defined outside of the feasible set, barrier methods must be initialized with a strictly feasible initial point $\mathbf{w}_0$.

Each barrier problem with barrier parameter μ is solved, e.g., using Newton iterations. Starting with an initial value for the barrier parameter μ and a strictly feasible initial point, the barrier method proceeds as summarized in Algorithm 1. For each iteration step, the associated log-barrier problem is solved using Newtons method and the obtained optimizers $\mathbf{w}_\mu^*$ are used as the initial values for the subsequent barrier problem. Each successive barrier problem reduces the barrier parameter by a factor $0 < \nu < 1$. The resulting sequence of optimizers $\mathbf{w}_\mu^*$ is commonly referred to as the central path and - for convex problems - converges to the optimal value of the original optimization problem.

Modified KKT conditions of the primal-dual interior-point method The PDIP algorithm implemented in IPOPT [95] is a variant of the interior-point method that solves a modified version of the Karush-Kuhn-Tucker (KKT) optimality conditions of the original problem (3.15). Compared to the log-barrier method, notable advantages of the PDIP algorithm are a better than linear convergence speed [98] and the ability to converge from poor starting points that are not strictly feasible without solving a feasibility problem [95].

Algorithm 1 Barrier method adapted from [98, Ch. 11.3]

Require: strictly feasible $\mathbf{w} > \mathbf{0}$, barrier parameter μ, sub-optimality ϵ, update factor $\nu \in (0, 1)$

while $m\mu \geq \epsilon$ **do**

Centering Step: compute $\mathbf{w}_\mu^*$ by solving (3.18) with Newtons steps using the initial point $\mathbf{w}$

Update: $\mathbf{w} \leftarrow \mathbf{w}_\mu^*$

Decrease μ*:* $\mu \leftarrow \nu\mu$

end while

To derive the modified KKT conditions, we compare the stationarity condition of the original problem (3.17) to the stationarity condition of the barrier problem (3.18). For the barrier method, the stationarity condition reads as

$$\mathbf{0} = \nabla_{\mathbf{w}} \mathcal{L}_\mu(\mathbf{w}, \lambda) = \nabla_{\mathbf{w}} \mathrm{f}(\mathbf{w}) - \mu \nabla_{\mathbf{w}} \sum_{i=1}^{m} \ln(w_i) - \nabla_{\mathbf{w}} \lambda^{\mathrm{T}} \mathrm{b}(\mathbf{w}) \tag{3.19}$$

$$= \nabla_{\mathbf{w}} \mathrm{f}(\mathbf{w}) - \mu \sum_{i=1}^{m} \frac{1}{w_i} - \nabla_{\mathbf{w}} \lambda^{\mathrm{T}} \mathrm{b}(\mathbf{w}). \tag{3.20}$$

By coefficient comparison to (3.17), we note that this corresponds to a particular choice of the dual variables z_i in the original stationarity condition (3.17), i.e.,

$$z_i = \mu \frac{1}{w_i},\ i = 1, \ldots, m \tag{3.21}$$

$$\Leftrightarrow z_i w_i = \mu,\ i = 1, \ldots, m. \tag{3.22}$$

Letting the barrier parameter go to zero $\mu \to 0$, we can see that this is a modified complementary slackness condition that converges to the exact complementary slackness condition in (3.17). Therefore, we can interpret the barrier method as a solution of modified KKT conditions [95], i.e.,

$$\begin{aligned} \nabla_{\mathbf{w}} \mathcal{L}(\mathbf{w}, \lambda, \mathbf{z}) = \nabla_{\mathbf{w}} \mathrm{f}(\mathbf{w}) - \nabla_{\mathbf{w}} \lambda^{\mathrm{T}} \mathrm{b}(\mathbf{w}) - \mathbf{z} = \mathbf{0}, &\quad \text{Stationarity} \\ \mathrm{diag}(\mathbf{z})\mathbf{w} - \mu \mathbf{1} = \mathbf{0}, &\quad \text{Complementary Slackness} \\ \mathbf{w} \geq \mathbf{0}, \quad \mathrm{b}(\mathbf{w}) = \mathbf{0}, &\quad \text{Primal Feasibility} \\ \mathbf{z} \geq \mathbf{0}. &\quad \text{Dual Feasibility} \end{aligned} \tag{3.23}$$

The primal-dual interior-point (PDIP) method implemented in IPOPT proceeds by performing Newton steps on the primal and dual variables, i.e.,

$$\mathbf{w}_{k+1} = \mathbf{w}_k + \alpha^{(k)} \Delta \mathbf{w}_k, \tag{3.24}$$

$$\lambda_{k+1} = \lambda_k + \alpha^{(k)} \Delta \lambda_k, \tag{3.25}$$

$$\mathbf{z}_{k+1} = \mathbf{z}_k + \alpha_{\mathbf{z}}^{(k)} \Delta \mathbf{z}_k, \tag{3.26}$$

where $\Delta\mathbf{w}_k,\ \Delta\lambda_k,\ \Delta\mathbf{z}_k$ are the step directions and $\alpha^{(k)}$ and $\alpha_{\mathbf{z}}^{(k)}$ are step sizes factors. Given an iterate $(\mathbf{w}_k,\ \lambda_k,\ \mathbf{z}_k)$, the step directions are computed by solving a linearization of the KKT conditions which is given by

$$\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}_k,\lambda_k,\mathbf{z}_k)+\nabla_{\mathbf{w}}^2\mathcal{L}(\mathbf{w}_k,\lambda_k,\mathbf{z}_k)\Delta\mathbf{w}+\nabla_{\mathbf{w}}\,\mathrm{b}(\mathbf{w}_k)\Delta\lambda-\Delta\mathbf{z}=\mathbf{0}, \tag{3.27}$$

$$\mathrm{b}(\mathbf{w}_k)+\nabla_{\mathbf{w}}\,\mathrm{b}^{\mathrm{T}}(\mathbf{w}_k)\Delta\mathbf{w}=\mathbf{0}, \tag{3.28}$$

$$\mathrm{diag}(\mathbf{z}_k)\mathbf{w}_k-\mu\mathbf{1}+\mathrm{diag}(\mathbf{z}_k)\Delta\mathbf{w}+\mathrm{diag}(\mathbf{w}_k)\Delta\mathbf{z}=\mathbf{0}. \tag{3.29}$$

Thus, to compute the step directions, the following equations must be solved [95],

$$\begin{bmatrix}\mathbf{L}_k & \mathbf{C}_k & -\mathbf{I}\\ \mathbf{C}_k^{\mathrm{T}} & \mathbf{0} & \mathbf{0}\\ \mathrm{diag}(\mathbf{z}_k) & \mathbf{0} & \mathrm{diag}(\mathbf{w}_k)\end{bmatrix}\begin{bmatrix}\Delta\mathbf{w}\\ \Delta\lambda\\ \Delta\mathbf{z}\end{bmatrix}=\begin{bmatrix}-\nabla_{\mathbf{w}}\mathcal{L}(\mathbf{w}_k,\lambda_k,\mathbf{z}_k)\\ -\,\mathrm{b}(\mathbf{w}_k)\\ -\,\mathrm{diag}(\mathbf{z}_k)\mathbf{w}_k+\mu\mathbf{1}\end{bmatrix}, \tag{3.30}$$

where $\mathbf{L}_k=\nabla_{\mathbf{w}}^2\mathcal{L}(\mathbf{w}_k,\lambda_k,\mathbf{z}_k)$ and $\mathbf{C}_k=\nabla_{\mathbf{w}}\,\mathrm{b}(\mathbf{w}_k)$ are the Hessian of the Lagrangian and the Jacobian matrix of the equality constraint at the kth iterate. In contrast to the barrier method, the PDIP method does not require that the optimization variables $\mathbf{z}$ are strictly larger than zero for all optimization steps since no logarithm functions must be evaluated. To satisfy the primal and dual feasibility constraints $\mathbf{w}\geq\mathbf{0}$ and $\mathbf{z}\geq\mathbf{0}$, respectively, IPOPT alternates between optimality updates as described above and feasibility updates which reduces primal and dual feasibility violations.

While the preceding equations capture the basic idea of the PDIP method, their algorithmic implementation poses important additional challenges. In particular, the step sizes $\alpha^{(k)}$ $\alpha_{\mathbf{z}}^{(k)}$ in each iteration step $_k$ crucially influence the local convergence speed and the global convergence behavior of the method, i.e., the ability to converge to a solution from any starting point [99]. Also, in practice, the system of equations (3.30) can be poorly scaled, making it necessary to carefully chose regularization schemes to obtain the step directions. The details on the algorithmic procedures of choosing these step sizes, as well as the solution of (3.30) in cases where the system of equations is poorly scaled or rank-deficient, is detailed in [95].

3.4 Alternative Numerical Methods for OCPs

The direct multiple shooting (DMS) method is an efficient and popular method for the solution of constrained optimal control problems. Here, we mention alternative numerical techniques used throughout the literature and point towards suitable survey papers. Numerical optimal control methods can be broadly classified as direct methods and indirect methods. A detailed discussion and comparison of direct and indirect methods can be found, e.g., in [100], [101].

Direct methods Direct methods of numerical optimal control discretize the control functions or the control *and* state functions to obtain a finite dimensional nonlinear optimization problem as discussed in the previous sections. Apart from the single and multiple shooting methods, the pseudospectral method [102] is a well-developed and popular direct solution method. The pseudospectral method is also referred to as global orthogonal collocation [100]: Instead of meshing the time horizon into separate intervals as in direct shooting, the control and state variables are approximated by global orthogonal polynomials which are defined over the entire time horizon of the optimal control problem. The differential equation constraints are solved by collocation. This means, that the derivatives of the global polynomials of the state trajectory are set to be equal to the differentials of the ODE at specific collocation points along the time horizon. The pseudospectral method is prominent in the area trajectory optimization for space applications, e.g., satellites trajectories or spacecraft reentry [102].

Indirect methods Indirect methods of numerical optimal control do not discretize the optimal control problem (OCP) itself. Instead, they solve first-order necessary optimality conditions of the OCP. These are given by Pontryagin's Maximum Principle (PMP), which is derived in detail, e.g., in [103]. The PMP leads to a boundary value problem which can be solved using methods like indirect single and multiple shooting. One reason why indirect methods are less prominent in the vehicle trajectory optimization literature is that they require an initial guess for the so-called adjoint variables of the boundary value problem. Compared to the initial guess required for direct methods, coming up with a guess for the adjoint variables can be significantly more difficult. As proposed in [103], the solution of the OCP obtained with a direct method could be used to find an initial guess for the adjoint variables of the boundary value problem of an indirect method.

Evolutionary Algorithms A third approach to numerically solving OCPs is based on so-called Evolutionary Algorithms (EA) that iteratively sample and improve a large set of candidate solutions. Popular algorithms of this type are Genetic Algorithms and Particle Swarm Optimization, see the survey paper [101]. Applying an EA algorithm to optimal control problems requires a finite dimensional (and preferably low dimensional) parameterization of the control variables. The candidate solutions are obtained by solving the ODE with candidate realizations of the control variables. A score function, which contains penalty terms for constraint violations, is used to measure the quality of the candidate solutions. Iteratively, the candidate solutions are altered, discarded, and mixed to improve the overall quality of the candidates. EA-based solutions to OCP problems tend to be easier to program than direct or indirect method. Additionally, EA itself does not require an initial guess for the

control or state variables. Thus, it can be used to compute an initial guess to use in a direct method. Disadvantages of the EA method of numerical optimal control, among others, are the necessity to use a low-dimensional control parameterization, the lack of a measure of (local) optimality, and the dependence on several tunable parameters which are unrelated to the OCP itself but have a significant influence on the quality of the obtained solutions [101].

Chapter 4

Optimal Control Formulation of Ground-Truth Criticality

The goal in this chapter is to define a labeling function that assigns a criticality c to a driving scenario represented by the scene tuple $\mathcal{S}$ (ref. (2.12)), i.e.,

$$c = \mathrm{c}(\mathcal{S}). \tag{4.1}$$

We propose that the following properties are desirable when designing or choosing a labeling function:

- *The labeling function should be easily interpretable to a human expert.* During the design, development, and use of the labeling function, the amount of detail and the application areas are likely to increase significantly. For example, a physics simulation could be extended from two to three-dimensional models of the world, or it could be extended to include intersection scenarios. To facilitate this development process, it should be easy to judge by a human expert whether the obtained labels appear to make sense. Another important reason for interpretable labels is the need to communicate and justify the label generation to non-experts, e.g., due to regulatory issues of the introduction of new driver assistance systems.

- *The labeling function should be based on the simulation of physics.* While this property appears obvious, it is not the only reasonable way to label the criticality of a driving scene. An alternative approach would be the labeling by human experts who could judge a driving scene as being threatening or not based on their subjective evaluation of, e.g., video recording or by driving in a simulator.

The subjectivity of this type of label generation can lead to what is commonly referred to as *label noise* in the machine learning literature (see, e.g., [104]). This is illustrated by the observation that, given the same driving scene, a group of human labelers can have vastly different reactions and experiences of threat. By generating simulation based labels, we avoid this source of label noise.

- *The labeling function should have as few tunable parameters as possible.* By tunable parameters, we mean parameters that are not determined by the physics of the driving scenes. A typical example of tunable parameters in ADAS is the a priori assumptions made about the driving behavior of the host vehicle driver and the obstacle. On the other hand, parameters like the mass of the ego vehicle are determined by the physics and are therefore not tunable. The number of tunable parameters directly influences the interpretability of the label and the ability to communicate and justify the labels to non-experts.

We propose the solution of an optimal control problem (OCP) as a labeling function $\mathrm{c}(\mathcal{S})$ for the ground-truth criticality labeling of longitudinal collision scenarios. The basic idea is to find an avoidance trajectory for the host vehicle that minimizes the maximum criticality during a prediction horizon T of a few seconds.

To obtain a criticality cost function for trajectory optimization, we first define a cost function for the criticality at a specific time $t \in [0, T]$, and then extend it to the entire prediction horizon $[0, T]$. We define the *instantaneous* criticality as a function $\mathrm{c}_t(\mathbf{s}(t), \mathbf{u}(t))$ which assigns a criticality to the state and control vectors $\mathbf{s}(t),\ \mathbf{u}(t)$ at a specific time $t \in [0, T]$ within the prediction horizon.

Without loss of generality, we first consider the norm of the acceleration vector as the instantaneous criticality measure, i.e.,

$$\mathrm{c}_t(\mathbf{s}(t), \mathbf{u}(t)) = \frac{\|\mathbf{a}(t)\|}{\mu g}, \quad \mathbf{a}(t) = \begin{bmatrix} a_{\mathrm{lon}}(t) \\ a_{\mathrm{lat}}(t) \end{bmatrix}, \tag{4.2}$$

where the longitudinal and lateral accelerations a_{lon} and a_{lat} are given by (2.25) and (2.26), and μ and g are the friction coefficient and the gravitational acceleration. This choice is motivated by the ease of interpretation of acceleration values relative to the gravitational acceleration g. In physics and aerospace engineering, the term *g-force*[1] is used to refer to multiples of the acceleration g due to gravity [106]. By dividing the acceleration norm by g, we obtain a normalized criticality measure where values greater than one indicate a highly critical driving state.

For trajectory optimization, our cost function has to assign a criticality to the state and control trajectories throughout the prediction horizon. We define the criticality

[1]The representation of the intensity of driving maneuvers in terms of g is also well established in popular culture: in the motor sport simulator *Project CARS* [105], the g-force vector is a prominent part of the user interface.

cost function for trajectories as

$$\mathrm{c}_{[0,T]}(\mathbf{s},\mathbf{u}) = \max_{t\in[0,T]} \mathrm{c}_t(\mathbf{s}(t),\mathbf{u}(t)), \tag{4.3}$$

where c_t is the instantaneous criticality function defined above and $\mathbf{s},\ \mathbf{u}$ are the state and control trajectories,

$$\mathbf{s} : [0,T] \to \mathbb{R}^8,\ \mathbf{u} : [0,T] \to \mathbb{R}^2. \tag{4.4}$$

The states and controls for the nonlinear single-track vehicle model are defined in Section 2.3.

Equations (4.2) and (4.3) define the criticality cost function of an avoidance maneuver as a function of the maximum dynamics of the host vehicle during the prediction horizon. This is an intuitive definition since it relates to the intensity or instability of the avoidance maneuver. For the criticality of an evasion maneuver, it doesn't matter whether the host vehicle becomes unstable at the start, middle, or end of the avoidance maneuver, only the peak instability is relevant. Owing to its straight-forward interpretability, the maximum value over time of the host vehicle dynamics has been used as a cost function, e.g., in [46] and [48].

We can now find the criticality $\mathrm{c}(\mathcal{S})$ of a collision scene $\mathcal{S}$ by minimizing the criticality function (4.3), i.e.,

$$\mathrm{c}(\mathcal{S}) = \min_{\mathbf{s},\mathbf{u}} \mathrm{c}_{[0,T]}(\mathbf{s},\mathbf{u}), \tag{4.5}$$

subject to a set of constraints on the state trajectory and control inputs:

- Lane keeping constraints, introduced in Chapter 4.1, enforce that the trajectory of the host vehicle does not veer off the lane.
- Anti-collision constraints, as detailed in Chapter 4.2, encode that the avoidance trajectory must be collision-free.
- Actor and tire limits constrain the maximum dynamics of an avoidance trajectory and are discussed in Chapter 4.3.

Assuming that the control inputs of a human driver will be less optimal than the control inputs computed by trajectory optimization, we can interpret the criticality value $\mathrm{c}(\mathcal{S})$ as a lower bound of the threat that a human driver is going to be exposed to in a particular driving situation $\mathcal{S}$.

Optimal control formulations of the collision avoidance problem have been successfully used in the past. The closest relationship to the formulation presented in this work is [46], where an OCP collision avoidance problem is solved to determine

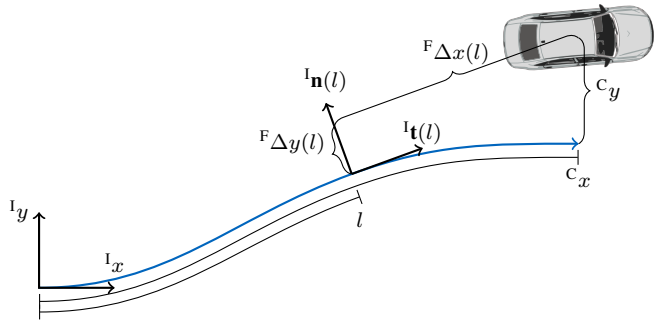


Figure 4.1: Lane keeping constraint with curvilinear path

whether to trigger automatic emergency braking. Said work shares the vehicle dynamics model and max-norm acceleration cost-function and treats the special case of straight road boundaries and a standing obstacle. A nuisance of that approach is the reliance on solving the OCP online to obtain criticality values, which imposes significant computational cost at runtime.

In [47], a hazard avoidance system based on model predictive control is proposed which gradually takes over control of the vehicle as the collision avoidance trajectory becomes harder to execute. It uses a complex threat model that combines the squared weighted norms constraint violation and control inputs. In [48], a two-level model predictive controller is proposed for the same application that combines path planning using a point mass model with decoupled longitudinal and lateral dynamics with path tracking using the nonlinear single-track model described in Chapter 2.

Avoidance trajectories are used in [49, 50] to approximate save reachable sets, which comprise the set of all collision-free trajectories. While the general setting of optimizing a cost function subject to lane keeping and collision avoidance is very comparable to this work, the cost functions is a weighted sum of maneuver execution time (which is not fixed) and control inputs. As such, it is neither intended nor suitable for criticality labeling as the optimized cost is hardly interpretable as a criticality. These works contain a treatment of the sensitivity analysis of the computed avoidance trajectories to estimate the influence of measurement errors on the safety of the trajectories.

4.1 Lane Keeping Constraints

We consider lane segments represented by smooth curves $l \mapsto {}^{\mathrm{I}}\mathbf{r}(l) \in \mathbb{R}^2$, where l is the path variable and ${}^{\mathrm{I}}\mathbf{r}(l)$ is referred to as the parameterization of the curve given in a road-fixed inertial coordinate system ${}^{\mathrm{I}}$. Figure 4.1 illustrates the following derivations.

To obtain a formulation of the lane keeping constraint, we have to find a Frenet

frame[2] for all $t \in [0, T]$ that has its origin at the projection of the COG of the host vehicle onto the path of the lane segment. We define the difference from the path coordinate $\mathbf{r}(l)$ to the position of the host vehicle as

$$\Delta\mathbf{r}(l) = {}^{\mathrm{I}}\mathbf{r}_{\mathrm{h}} - \mathbf{r}(l), \tag{4.6}$$

where ${}^{\mathrm{I}}\mathbf{r}_{\mathrm{h}}$ is the position of the COG of the host vehicle. This difference vector can be expressed in the coordinate system of a Frenet frame at the arc length l as

$$ {}^{\mathrm{F}}x(l) = {}^{\mathrm{I}}\mathbf{t}(l)^{\mathrm{T}}\Delta\mathbf{r}(l), \tag{4.7}$$
$$ {}^{\mathrm{F}}y(l) = {}^{\mathrm{I}}\mathbf{n}(l)^{\mathrm{T}}\Delta\mathbf{r}(l). \tag{4.8}$$

By inspection of Figure 4.1, we can deduce that the COG position expressed in the Frenet frame at the projection of ${}^{\mathrm{I}}\mathbf{r}_{\mathrm{h}}$ onto the path would be orthogonal to the normal vector of the path $\mathbf{n}(l)$. Thus, we obtain an implicit equation for the aligned coordinate of the host vehicle $l = {}^{\mathrm{C}}x$, i.e.,

$$ {}^{\mathrm{F}}x(l) = {}^{\mathrm{I}}\mathbf{t}(l)^{\mathrm{T}}\Delta\mathbf{r}(l) = 0. \tag{4.9}$$

We can determine the Frenet frame by adding the path variable l as an auxiliary variable to the OCP (4.5) together with the constraint (4.9). The lane keeping constraints now read as

$$\begin{aligned} {}^{\mathrm{F}}x(l) &= 0, \\ -B_{\mathrm{r}} + B_{\Delta} \leq {}^{\mathrm{F}}y(l) &\leq B_{\mathrm{l}} - B_{\Delta}, \end{aligned} \tag{4.10}$$

where $B_{\mathrm{l}},\ B_{\mathrm{r}}$ are the maximum distances to the left and right of the lane center, and B_{Δ} is a safety margin, which we chose as half of the host vehicle width, $B_{\Delta} = 0.5W_{\mathrm{h}}$. An alternative to constraining the host vehicle COG position would be a constraint on all corners of the host vehicle body. This would, however, lead to an increase in the number of constraints by a factor of the number of considered corners.

While this formulation of the lane keeping is applicable for any smooth path, we have implemented it for lane keeping on clothoidal lanes. Refer to Chapter 2.1 for the formulae of the curve $\mathbf{r}(l)$ and the associated Frenet basis ${}^{\mathrm{I}}\mathbf{t}(l), {}^{\mathrm{I}}\mathbf{n}(l)$.

Curvilinear coordinates have previously been used for the prediction and tracking of obstacles in lane-following scenarios [73]. The authors of [107, 108] use a so-called path-parametric system reformulation, i.e., the differential equations of the host vehicle are reformulated using the path variable l as an independent variable instead of the time t. As in (4.10), this has the advantage of a simple formulation of the lane keeping constraint in terms of upper and lower bounds on the offset to the lane center.

[2] The Frenet frame is an orthonormal basis consisting of the unit-norm tangent and normal vectors of the path $\mathbf{r}(l)$ as introduced in Chapter 2.1.

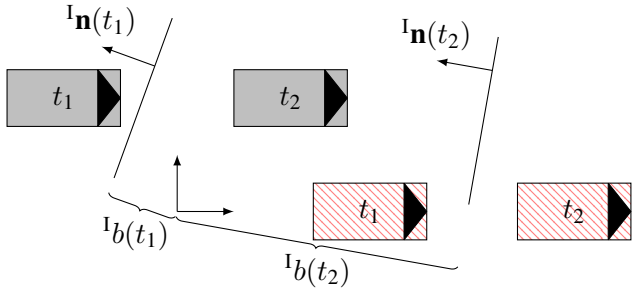


Figure 4.2: Separating hyperplane and auxiliary variables for collision avoidance

Further, the path variable l does not have to be added to the OCP in those approaches. However, these path-parametric formulations only treat static obstacles. The author is not aware of any path-parametric OCP formulations that combine curvilinear lane keeping with the treatment of dynamic obstacles. In [50, Ch. 2.2], different road geometries are represented as intersections of level sets which are used as constraints in the OCP. This approach is comparable to the collision-avoidance formulation of [109] which uses unions of polyhedrons to describe non-convex obstacle geometries.

4.2 Anti-Collision Constraints

To find collision-free trajectories, we must formulate constraints that ensure that the body geometry of the host vehicle never overlaps with the obstacle geometry during the prediction interval. In [46, 49, 107], the obstacle has a fixed position with respect to the road. In those cases, the anti-collision constraint is merely a lower or upper bound on the COG position along the lane. For a straight lane model, the authors of [110] formulate the collision avoidance constraint for a time-optimal lane-change as a constraint on the COG distance to the obstacle at $t = T$. While straight-forward for time-optimal trajectories, the approach is not applicable to find trajectories with minimal dynamics, since the constraint only applies to the final state instead of the whole time interval. In [111, 112], the host vehicle geometry is approximated as a circle, whereas a weighted p-Norm is used to compute distances from the host vehicle to the obstacle centers. The weights of the p-norm and the chosen p itself are used to model different obstacle geometries. This approach is closely related to the artificial potential field methods used for trajectory planning in robotics [113].

Separating Hyperplane Constraint Here, we propose a collision avoidance constraint formulation that applies to convex polytope models of obstacles. Denoting the sets of points belonging to the host and vehicle body geometries as $\mathcal{B}_{\mathrm{h}} : \ t \mapsto \mathcal{B}_{\mathrm{h}}(t) \subset \mathbb{R}^2$

and $\mathcal{B}_{\text{obs}}: \ t \mapsto \mathcal{B}_{\text{obs}}(t) \subset \mathbb{R}^2$, we can write the anti-collision constraint as

$$\mathcal{B}_{\text{h}}(t) \cap \mathcal{B}_{\text{obs}}(t) = \emptyset \ \forall t \in [0, T]. \tag{4.11}$$

We approximate the geometries of the host and obstacle by closed convex polytopes, which we write as the convex hull of the set of corner points, i.e.,

$$\mathcal{B}_{\text{h}} = \text{chull}\{{}^{\text{I}}\mathbf{c}_{\text{h},1}, \ldots, {}^{\text{I}}\mathbf{c}_{\text{h},N_{\text{h}}}\}, \tag{4.12}$$

$$\mathcal{B}_{\text{obs}} = \text{chull}\{{}^{\text{I}}\mathbf{c}_{\text{obs},1}, \ldots, {}^{\text{I}}\mathbf{c}_{\text{obs},N_{\text{obs}}}\}, \tag{4.13}$$

where $N_{\text{h}}, \ N_{\text{obs}}$ are the number of points used to describe the geometries and chull is the convex hull operator which assigns to a set of points the smallest convex set that contains them [98].

For defining the geometry of the restricted area, we assume without loss of generality, that the obstacle drives on the right-hand side of the center line. Let ${}^{\text{I}}\mathbf{c}_{1,\text{obs}} = [x_{1,\text{obs}}, y_{1,\text{obs}}]^{\text{T}}$ denote the projection of the inner corner of the rear of the obstacle onto the center line. Further, let ${}^{\text{I}}\mathbf{n}_1$ and ${}^{\text{I}}\mathbf{t}_1$ be the normal and tangential vectors of the lane center path at ${}^{\text{I}}\mathbf{c}_{1,\text{obs}}$. In road-fixed coordinates, this point is denoted as ${}^{\text{I}}\mathbf{c}_{1,\text{obs}}$ [3] . The geometric definition of the restricted area can now be stated as

$$\begin{aligned}
\mathcal{B}_{\text{obs}} &= \text{chull}\left\{{}^{\text{I}}\mathbf{c}_{1,\text{obs}}, {}^{\text{I}}\mathbf{c}_{2,\text{obs}}, {}^{\text{I}}\mathbf{c}_{3,\text{obs}}, {}^{\text{I}}\mathbf{c}_{4,\text{obs}}\right\}, \\
{}^{\text{I}}\mathbf{c}_{2,\text{obs}} &= {}^{\text{I}}\mathbf{c}_{1,\text{obs}} - B_{\text{r}}\, {}^{\text{I}}\mathbf{n}_1, \\
{}^{\text{I}}\mathbf{c}_{3,\text{obs}} &= {}^{\text{I}}\mathbf{c}_{1,\text{obs}} + L\, {}^{\text{I}}\mathbf{t}_1, \\
{}^{\text{I}}\mathbf{c}_{4,\text{obs}} &= {}^{\text{I}}\mathbf{c}_{2,\text{obs}} + L\, {}^{\text{I}}\mathbf{t}_1,
\end{aligned} \tag{4.14}$$

where L is the length of the obstacle and chull is the convex hull as above. The position of the obstacle over time is given by the constant acceleration and lane-following assumption detailed in Chapter 2.

Rigid-body transformations give the corner positions of the host vehicle over time, i.e.,

$${}^{\text{I}}\mathbf{c}_{\text{h},i}(t) = \mathbf{R}({}^{\text{I}}\psi(t))\, {}^{\text{V}}\mathbf{c}_{\text{h},i} + {}^{\text{I}}\mathbf{r}_{\text{h}}(t), \tag{4.15}$$

where ${}^{\text{V}}\mathbf{c}_{\text{h},i}$ are the corners points of the host vehicle, and ${}^{\text{I}}\mathbf{r}_{\text{h}}(t)$ and ${}^{\text{I}}\psi(t)$ are the position and yaw angle of the host vehicle at time t.

Since the obstacle and host vehicle geometries are assumed to be convex, we can employ the separating hyperplane theorem (SHT) [98] to obtain an algebraic formulation of the anti-collision constraints. The SHT states that if two closed convex

[3] The transformation between road fixed coordinates $^{\text{I}}$ and clothoidal coordinates $^{\text{C}}$ used in this definition given in Chapter 2.1.

sets $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^N$ are disjoint then there exists a hyperplane given by a scalar b and normal vector $\mathbf{n} \in \mathbb{R}^N$ such that

$$\begin{aligned} {}^{\mathrm{I}}\mathbf{n}^{\mathrm{T}}\mathbf{a} - {}^{\mathrm{I}}b \leq 0 \; \forall \mathbf{a} \in \mathcal{A}, \\ {}^{\mathrm{I}}\mathbf{n}^{\mathrm{T}}\mathbf{b} - {}^{\mathrm{I}}b \geq 0 \; \forall \mathbf{b} \in \mathcal{B}. \end{aligned} \tag{4.16}$$

In the two-dimensional case, illustrated in Figure 4.2, we can parameterize the normal vector by its polar angle ${}^{\mathrm{I}}\phi$, i.e.,

$${}^{\mathrm{I}}\mathbf{n}_{\phi} = \begin{bmatrix} \cos\left({}^{\mathrm{I}}\phi\right) & \sin\left({}^{\mathrm{I}}\phi\right) \end{bmatrix}^{\mathrm{T}}. \tag{4.17}$$

Since we define the body geometries as convex polytopes, we can write the anti-collision constraints as $N_{\mathrm{h}} + N_{\mathrm{obs}}$ inequality constraints for the OCP (4.5), i.e.,

$$\begin{aligned} {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}\mathbf{c}_{\mathrm{h},i} - {}^{\mathrm{I}}b \leq 0 \; \forall \mathbf{c}_{\mathrm{h},i} \in \mathcal{B}_{\mathrm{h}}, \\ {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}\mathbf{c}_{\mathrm{obs},j} - {}^{\mathrm{I}}b \geq 0 \; \forall \mathbf{c}_{\mathrm{obs},j} \in \mathcal{B}_{\mathrm{obs}}. \end{aligned} \tag{4.18}$$

As in the case of the path variable l for the lane keeping constraint, the hyperplane angle ${}^{\mathrm{I}}\phi$ and scalar ${}^{\mathrm{I}}b$ become auxiliary variables.

An alternative formulation of the anti-collision constraint based on Farkas lemma is proposed in [109]. Further, the authors of [109] propose a generalization to non-convex obstacles by decomposition into unions of convex sets. The convex sets $\mathcal{A}, \mathcal{B} \subset \mathbb{R}^2$ are defined by the inequalities

$$\mathcal{A} = \left\{ \mathbf{r} \in \mathbb{R}^2 : \mathbf{A}\mathbf{r} \leq \mathbf{b} \right\}, \quad \mathbf{A} \in \mathbb{R}^{N_A \times 2}, \mathbf{b} \in \mathbb{R}^{N_A} \tag{4.19}$$

$$\mathcal{B} = \left\{ \mathbf{r} \in \mathbb{R}^2 : \mathbf{C}\mathbf{r} \leq \mathbf{d} \right\}, \quad \mathbf{C} \in \mathbb{R}^{N_B \times 2}, \mathbf{d} \in \mathbb{R}^{N_B}. \tag{4.20}$$

If $\mathcal{A}$ and $\mathcal{B}$ are disjoint, there is no solution to the system of inequalities,

$$\begin{bmatrix} \mathbf{A} \\ \mathbf{C} \end{bmatrix} \mathbf{r} \leq \begin{bmatrix} \mathbf{b} \\ \mathbf{d} \end{bmatrix}. \tag{4.21}$$

According to Farkas lemma, this is equivalent to the existence of a vector $\mathbf{w} \in \mathbb{R}^{N_A + N_B}$ that solves the system of equations [109, Proposition 1]

$$\mathbf{w} \geq \mathbf{0}, \quad \begin{bmatrix} \mathbf{A} \\ \mathbf{C} \end{bmatrix}^{\mathrm{T}} \mathbf{w} = \mathbf{0}, \quad \begin{bmatrix} \mathbf{b} \\ \mathbf{d} \end{bmatrix}^{\mathrm{T}} \mathbf{w} < 0. \tag{4.22}$$

To be able to optimize over a closed set, the authors of [109] replace the strict inequality in (4.22) by the non-strict inequality $\begin{bmatrix} \mathbf{b}^{\mathrm{T}} & \mathbf{d}^{\mathrm{T}} \end{bmatrix} \mathbf{w} \leq -\epsilon$, where $\epsilon > 0$ is a small fudge factor. This set of inequalities is then added as constraints to the OCP and the vector $\mathbf{w}$ becomes an auxiliary variable.

The hyperplane formulation in (4.18) adds two control variables - ${}^{\mathrm{I}}\phi$ and ${}^{\mathrm{I}}b$ - and $N_{\mathrm{h}} + N_{\mathrm{obs}}$ constraints. With $N_A = N_{\mathrm{h}}, N_B = N_{\mathrm{obs}}$, the formulation based on Farkas lemma adds $N_{\mathrm{h}} + N_{\mathrm{obs}}$ auxiliary variables and $N_{\mathrm{h}} + N_{\mathrm{obs}} + 3$ constraints (see (4.22)) as well as a tunable fudge factor ϵ. Thus, the proposed SHT formulation compares favorably to the formulation in [109].

4.3 Input and State Constraints

The braking and steering actors are limited in the rate with which they can build up the braking force and steering angle, which significantly influences the feasibility of collision avoidance. These actor limitations are reflected by the box constraints

$$|u_1| \leq u_{1,\max}, \quad |u_2| \leq u_{2,\max}. \tag{4.23}$$

We constrain the steering rate to $u_{2,\max} = -u_{2,\min} = \frac{2\pi}{15}\text{rad/s}$. This choice amounts to about one turn of the steering wheel per seconds with a steering ratio[4] of 15 : 1. We set the bounds on the force rate to $u_{1,\min} = -\frac{\mu m g}{0.2}$ and $u_{1,\max} = -5u_{1,\min}$, which allows a deceleration of μg in two hundred milliseconds. The value of $u_{1,\max}$ corresponds to a sudden release of the brake. We bound the total tangential tire force from below by

$$F_\text{t} \geq F_{\text{t},\min}, \quad F_{\text{t},\min} = -\mu m g, \tag{4.24}$$

which allows a maximum deceleration of μg.

We don't impose an upper bound on the tangential tire force, since we only consider braking-and-steering evasion maneuvers where an upper bound would never be tight. Similarly, we also don't impose upper and lower bounds on the steering angle. This allows us to get a feasible solution and criticality label for small velocities where we would otherwise not be able to obtain a label. We discuss the trade-off between actor and state constraints and feasibility in Section 4.5.1, where we propose alternatives to directly constraining the control or states of the host vehicle.

These constraint values are a guess on what a modern braking and steering system might achieve, and, in practice, we would have to adapt them to a concrete vehicle and use case. A valuable extension of the actor constraints would be a model for the dead time of the actors.

4.4 Initial and Final State Constraints

The initialization proposed here assumes stationary lane-following by the host vehicle. This assumption holds true whenever a safety maneuver is necessary due to the inattentiveness of the driver or due to an obstacle that suddenly becomes visible, e.g., due to prior visual occlusion. By using implicit equations for yaw and steering angles, we let the OCP solver find a suitable initial state instead of providing plausible initial values for all states. For straight lanes, we position the origin of the inertial coordinate system $^\text{I}$ at the projection of the COG onto the lane center. The initial state constraint writes as

$$\mathbf{s}(0) = \mathbf{s}_0, \quad \mathbf{s}_0 = \begin{bmatrix} 0 & {}^\text{I}y_0 & v_0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}^\text{T}. \tag{4.25}$$

[4]The steering ratio relates the turning angle of the steering wheel to the resulting turning angle of the tire. In modern vehicles, it is dynamically adapted to the speed of the vehicle and other factors.

For curvilinear lane segments, we fix the origin of $^{\mathrm{I}}$ to the origin of the curve, see Chapter 2.1. Choosing a different origin for $^{\mathrm{I}}$, e.g., at the projection of the COG onto the curve, would make it necessary to transform the parameterization of the curve to that coordinate system - an avoidable nuisance.

First, we compute the initial value of the yaw rate, taking into account the speed of the vehicle as well as the lane curvature. Given a parameterization of $\mathbf{r}(l)$ of the lane segment, the curvature is given by the Frenet equation for planar curves [77, Ch. 2.5],

$$^{\mathrm{I}}\mathbf{t}' = c(l)\,{}^{\mathrm{I}}\mathbf{n}(l). \tag{4.26}$$

For clothoids, the signed curvature is a linear function of the arc length, i.e.,

$$c = \kappa l, \tag{4.27}$$

with a constant rate of change of the curvature κ, ref. Chapter 2.1. Note that the sign of the curvature depends on the sign of the arc length variable l.

We denote by l_0 the arc length of the clothoid that corresponds to the projection of the host COG onto the lane center, see Figure 4.1. $^{\mathrm{C}}y_0$ denotes the initial lateral offset of the host vehicle relative to the lane center. Since we assume that the host vehicle follows the lane at a constant velocity at the beginning of the simulation, the rate of change of the body slip angle is negligible compared to the yaw rate, i.e., $\dot{\beta} << \omega$. Thus, we can determine the initial value of the yaw rate from the initial curvature of the trajectory of the host vehicle, $\tilde{c_0}$, and its velocity, i.e.,

$$\omega_0 = v_0 \tilde{c_0}. \tag{4.28}$$

The curvature $\tilde{c}_0$ of the host trajectory is the inverse of the initial turning radius, i.e.,

$$\tilde{c}_0 = \frac{1}{r_0}. \tag{4.29}$$

Under the simplifying assumption that the initial velocity vector is parallel to the lane center marking, we can compute the initial turning radius r_0 as,

$$r_0 = \frac{1}{c_0} + {}^{\mathrm{C}}y_0, \tag{4.30}$$

where $^{\mathrm{C}}y_0$ is the offset of the COG from the lane center.

We can now write a first set of initial state constraints,

$$\mathbf{1}_{\mathrm{eq}}\mathbf{s}(0) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix} \mathbf{s}(0) = \begin{bmatrix} {}^{\mathrm{I}}x_0 \\ {}^{\mathrm{I}}y_0 \\ v_0 \\ \omega_0 \\ F_{\mathrm{tot},0} \end{bmatrix} = \mathbf{s}_{\mathrm{eq}}. \tag{4.31}$$

The linear state constraints in (4.31) only initialize five out of eight states. Thus, we need three more implicit equations for the remaining states, i.e., the body slip angle $\beta(0)$, the yaw angle $\psi(0)$, as well as the steering angle $\delta(0)$. Due to the velocity vector being parallel to the curve, the heading angle ${}^{\mathrm{I}}\theta = {}^{\mathrm{I}}\psi_{\mathrm{h}} + \beta$ is equal to the angle ${}^{\mathrm{I}}\tau_0$ of the curve tangent,

$$ {}^{\mathrm{I}}\psi_{\mathrm{h}}(0) + \beta(0) = {}^{\mathrm{I}}\tau_0, \quad {}^{\mathrm{I}}\tau_0 = {}^{\mathrm{I}}\tau(l_0), \tag{4.32} $$

where the tangent angle is given by

$$ {}^{\mathrm{I}}\tau(l) = \arctan\left(\frac{[{}^{\mathrm{I}}\mathbf{t}]_y}{[{}^{\mathrm{I}}\mathbf{t}]_x}(l)\right), \tag{4.33} $$

with the x and y components of the tangent written as $[{}^{\mathrm{I}}\mathbf{t}]_x$ and $[{}^{\mathrm{I}}\mathbf{t}]_x$, respectively. For the special case of clothoidal lane segments, the tangent angle is given in (2.3).

Due to the nonlinearity of the ODE of the host vehicle, we cannot give explicit equations for the initial body slip angle β_0 and steering angle δ_0. Instead, we constrain the states $\beta(0)$ and $\delta(0)$ by implicit equations. To this end, we approximate the body slip angle and yaw rate as stationary. From the ODE equations for the single-track model (2.39), we can give the resulting constraints as

$$ \begin{aligned} \dot{\beta}(0) = 0 &= \frac{a_{\mathrm{lat}}}{v} - \omega, \\ \dot{\omega}(0) = 0 &= \frac{1}{I}(F_{\mathrm{s,f}}\cos(\delta)l_f + F_{\mathrm{t,f}}\sin(\delta)l_f - F_{\mathrm{s,r}}l_r). \end{aligned} \tag{4.34} $$

We collect equations (4.31), (4.32) and (4.34) in the nonlinear initial state constraint

$$ \mathbf{r}_{\mathrm{init}}(\mathbf{s}(0)) = \begin{bmatrix} \mathbf{1}_{\mathrm{eq}}\mathbf{s}(0) - \mathbf{s}_{\mathrm{eq}} \\ {}^{\mathrm{I}}\psi_{\mathrm{h}}(0) + \beta(0) - {}^{\mathrm{I}}\tau_0 \\ \frac{a_{\mathrm{lat}}}{v}(0) - \omega(0) \\ \frac{1}{I}(F_{\mathrm{s,f}}(0)\cos(\delta(0))l_f + F_{\mathrm{t,f}}(0)\sin(\delta(0))l_f - F_{\mathrm{s,r}}(0)l_r) \end{bmatrix} = \mathbf{0}, \tag{4.35} $$

which imposes eight constraints for eight variables in $\mathbf{s}(0)$ leading to well-defined initial states.

Final state constraints At the end of the prediction horizon $t = T$ we want the vehicle body to be aligned with the driving direction of the lane to allow a safe continuation of the evasion trajectory for $t > T$. This is encoded by a constraint on the yaw angle and the arc length variable at the time $t = T$,

$$ \begin{aligned} {}^{\mathrm{I}}\psi(T) - {}^{\mathrm{I}}\tau(l(T)) &\leq \psi_{\mathrm{end}}, \\ {}^{\mathrm{I}}\psi(T) - {}^{\mathrm{I}}\tau(l(T)) &\geq -\psi_{\mathrm{end}}, \end{aligned} \tag{4.36} $$

where ${}^{\mathrm{I}}\tau(l)$ is given in (4.33) and ψ_{end} is an upper bound on the deviation from the lane direction. We have chosen $\psi_{\text{end}} = 10^\circ$ throughout our simulations, see Table 4.2. The host vehicle is forced to end on the side of the lane marking that is not occupied by the obstacle. Let ${}^{\mathrm{C}}y_{\text{obs},0} \neq 0$ denote the y coordinate of the center of the rear end of the obstacle, which we assume to be unequal to zero. We can then express the lane change constraint using a case distinction,

$$\begin{aligned} {}^{\mathrm{C}}y(T) &= \geq \frac{W_{\mathrm{h}}}{2} && \text{if } {}^{\mathrm{C}}y_{\text{obs},0} < 0, \\ {}^{\mathrm{C}}y(T) &= \leq -\frac{W_{\mathrm{h}}}{2} && \text{if } {}^{\mathrm{C}}y_{\text{obs},0} > 0. \end{aligned} \tag{4.37}$$

A more compact form without the case distinction is the inequality

$${}^{\mathrm{C}}y(T)\,{}^{\mathrm{C}}y_{\text{obs},0} \leq -\frac{W_{\mathrm{h}}}{2}|\,{}^{\mathrm{C}}y_{\text{obs},0}|. \tag{4.38}$$

In the criticality OCP with curved lane segments, we use the Frenet frame (see, 2.1) to express the lane-change constraint. Thus, we have

$$\begin{aligned} {}^{\mathrm{F}}x(l(T)) &= 0, \\ {}^{\mathrm{F}}y(l(T))\,{}^{\mathrm{C}}y_{\text{obs},0} &\leq -\frac{W_{\mathrm{h}}}{2}|\,{}^{\mathrm{C}}y_{\text{obs},0}|. \end{aligned} \tag{4.39}$$

We collect the final state constraints (4.36) and (4.39) for curved lanes segments in the inequality

$$\mathbf{r}_{\text{end}}(\mathbf{s}(T)) = \begin{bmatrix} {}^{\mathrm{F}}x(l(T)) \\ -{}^{\mathrm{F}}x(l(T)) \\ {}^{\mathrm{I}}\psi(T) - {}^{\mathrm{I}}\tau(l(T)) - \psi_{\text{end}} \\ {}^{\mathrm{I}}\tau(l(T)) - {}^{\mathrm{I}}\psi(T) - \psi_{\text{end}} \\ {}^{\mathrm{F}}y(l(T))\,{}^{\mathrm{C}}y_{\text{obs},0} + \frac{W_{\mathrm{h}}}{2}|\,{}^{\mathrm{C}}y_{\text{obs},0}| \end{bmatrix} \leq \mathbf{0}. \tag{4.40}$$

4.5 Optimal Control Problem Formulations

Having derived lane keeping, anti-collision, state, and control constraints, we can now state the labeling OCP. We formulate the OCPs straight and curvilinear lane segments, identify the path constraints, and reformulate the OCPs for numerical optimization. In Section 4.5.1, we discuss an alternative criticality OCP with relaxed control constraints.

For straight lane segments, we collect Equations (2.39), (4.25), (4.36), (4.39), (4.18), and (4.10) in the OCP formulation. We use an inertial coordinate system ${}^{\mathrm{I}}$ with the x-Axis aligned with the lane direction, so the tangent angle of the lane is always

zero, $\tau = 0$. The origin is located at the projection of the COG onto the lane center, see Figure 2.2. The OCP becomes

$$
\begin{aligned}
&\min_{\substack{\mathbf{s}(t),\mathbf{u}(t),\\ b(t),\phi(t),\\ \forall t\in[0,T]}} \max_{t\in[0,T]} \mathrm{c}_t(\mathbf{s}(t),\mathbf{u}(t)),\ \text{s.t.} \\
&\qquad \dot{\mathbf{s}}(t) - f(\mathbf{s}(t),\mathbf{u}(t)) = \mathbf{0}, && \text{ODE constraints, see (2.39)} \\
&\qquad \mathbf{s}(0) - \mathbf{s}_0 = \mathbf{0}, && \text{initial state constraints, see (4.35)} \\
&\qquad {}^{\mathrm{I}}\psi(T) - \psi_{\text{end}} \leq 0, && \text{final state constraints, see (4.40)} \\
&\qquad -{}^{\mathrm{I}}\psi(T) - \psi_{\text{end}} \leq 0, \\
&{}^{\mathrm{I}}y(T)\,{}^{\mathrm{I}}y_{\text{obs},0} + \frac{W_{\mathrm{h}}}{2}|{}^{\mathrm{I}}y_{\text{obs},0}| \leq 0, \\
&\qquad {}^{\mathrm{I}}y(l(t)) - B_{\mathrm{l}} + B_{\Delta} \leq 0, && \text{path constraints} \\
&\qquad -B_{\mathrm{r}} + B_{\Delta} - {}^{\mathrm{I}}y(l(t)) \leq 0, \\
&\qquad {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{h},i} - {}^{\mathrm{I}}b(t) \leq 0,\ \forall i = 1,\ldots,N_{\mathrm{h}}, \\
&\qquad {}^{\mathrm{I}}b(t) - {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\text{obs},j} \leq 0,\ \forall j = 1,\ldots,N_{\text{obs}}, \\
&\qquad F_{\text{t,min}} - F_{\mathrm{t}}(t) \leq 0, \\
&\qquad u_1(t) - u_{1,\text{max}} \leq 0, \\
&\qquad u_{1,\text{min}} - u_1(t) \leq 0, \\
&\qquad u_2(t) - u_{2,\text{max}} \leq 0 \\
&\qquad u_{2,\text{min}} - u_2(t) \leq 0\ \forall t \in [0,T]. && (4.41)
\end{aligned}
$$

Compared to the OCP for straight lanes, the curvilinear formulation requires the auxiliary optimization variable l, i.e., the arc length along the lane segment, and the expression of the constraints in two different coordinate systems ${}^{\mathrm{I}}$ and ${}^{\mathrm{C}}$. Also, the initial state constraints are now given by the nonlinear equation $\mathbf{r}_{\text{init}}(\mathbf{s}(0)) = \mathbf{0}$ as defined in (4.35). Collecting equations (2.39), (4.35), (4.40), (4.9), (4.10), and (4.18),

the collision avoidance OCP for curvilinear lane segments writes as

$$
\begin{aligned}
&\min_{\substack{\mathbf{s}(t),\mathbf{u}(t),\\ l(t),b(t),\phi(t),\\ \forall t\in[0,T]}} \ \max_{t\in[0,T]} \mathrm{c}_t(\mathbf{s}(t),\mathbf{u}(t)),\ \text{s.t.}\\
&\qquad \dot{\mathbf{s}}(t) - f(\mathbf{s}(t),\mathbf{u}(t)) = \mathbf{0}, && \text{ODE constraints, see (2.39)}\\
&\qquad \mathbf{r}_{\text{init}}(\mathbf{s}(0)) = \mathbf{0}, && \text{initial state constraints, see (4.35)}\\
&\qquad \mathbf{r}_{\text{end}}(\mathbf{s}(T)) \leq \mathbf{0}, && \text{final state constraints, see (4.40)}\\
&\qquad {}^{\mathrm{F}}x(l(t)) = 0, && \text{path constraints}\\
&\qquad {}^{\mathrm{F}}y(l(t)) - B_{\mathrm{l}} + B_{\Delta} \leq 0,\\
&\qquad -B_{\mathrm{r}} + B_{\Delta} - {}^{\mathrm{F}}y(l(t)) \leq 0,\\
&\qquad {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{h},i} - {}^{\mathrm{I}}b(t) \leq 0,\ \forall i = 1,\ldots,N_{\mathrm{h}},\\
&\qquad {}^{\mathrm{I}}b(t) - {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{obs},j} \leq 0,\ \forall j = 1,\ldots,N_{\mathrm{obs}},\\
&\qquad F_{\mathrm{t,min}} - F_{\mathrm{t}}(t) \leq 0,\\
&\qquad u_1(t) - u_{1,\mathrm{max}} \leq 0,\\
&\qquad u_{1,\mathrm{min}} - u_1(t) \leq 0,\\
&\qquad u_2(t) - u_{2,\mathrm{max}} \leq 0,\\
&\qquad u_{2,\mathrm{min}} - u_2(t) \leq 0\ \forall t \in [0,T]. && (4.42)
\end{aligned}
$$

In the following, we concentrate on OCP (4.42) for curvilinear lane segments and derive a compact notation of the OCP that is suitable for numerical optimization. For the straight-lane case, the same reformulation steps apply. First, as in [46, 48], we restate the minimization of the maximum value of the criticality over time as the minimization of an upper bound[5] z, i.e.,

$$
\begin{aligned}
&\mathrm{c}(\mathcal{S}) = \sqrt{z_{\text{opt}}},\\
&z_{\text{opt}} = \min_{\substack{z,\\ \mathbf{s}(t),\mathbf{u}(t),\\ l(t),b(t),\phi(t),\\ \forall t\in[0,T]}} \ z \quad \text{s.t. } \mathrm{c}_t^2(\mathbf{s}(t),\mathbf{u}(t)) \leq z\ \forall t \in [0,T]. \qquad (4.43)
\end{aligned}
$$

Optimizing an upper-bound of the cost function instead of the maximum-value of the cost function yields smooth gradients and Hessians during numerical optimization. Note that we have replaced the instantaneous criticality $\mathrm{c}_t(\mathbf{s}(t),\mathbf{u}(t)$ by its squared value. This also serves to obtain smooth gradients and Hessians. Consider, e.g., the acceleration norm criticality in Equation (4.2), whose gradient is discontinuous at zero. Optimizing the squared norm fixes this problem. A precondition for optimizing the squared criticality instead of the criticality is that the criticality is greater or equal

[5] The minimization $\min_z z$ can be cast as an instance of the general OCP problem (3.1) by writing it as $\min_{z(t)} \int_0^T z(t), \text{s.t. } \dot{z}(t) = 0$.

to zero, which is obviously satisfied for the criticality considered here. To recover the criticality value, we take the square-root of the optimized upper-bound z_{opt}.

Next, for conciseness, we subsume the state and auxiliary variables in an augmented state vector

$$\tilde{\mathbf{s}} = \begin{bmatrix} \mathbf{s}^{\mathrm{T}} & {}^{\mathrm{I}}\phi & {}^{\mathrm{I}}b & l \end{bmatrix}^{\mathrm{T}}, \tag{4.44}$$

and collect the initial, final, and path constraints. The path constraints read as

$$\mathbf{g}_{\text{path}}(\tilde{\mathbf{s}}(t), \mathbf{u}(t)) = \begin{bmatrix} {}^{\mathrm{F}}x(l(t)) \\ -{}^{\mathrm{F}}x(l(t)) \\ {}^{\mathrm{F}}y(l(t)) - B_{\mathrm{l}} + B_{\Delta} \\ -B_{\mathrm{r}} + B_{\Delta} - {}^{\mathrm{F}}y(l(t)) \\ {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{h},1} - {}^{\mathrm{I}}b(t) \\ \vdots \\ {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{h},N_{\mathrm{h}}} - {}^{\mathrm{I}}b(t) \\ {}^{\mathrm{I}}b(t) - {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{obs},1} \\ \vdots \\ {}^{\mathrm{I}}b(t) - {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{obs},N_{\mathrm{obs}}} \\ F_{\mathrm{t,min}} - F_{\mathrm{t}}(t) \\ u_1(t) - u_{1,\max} \\ u_{1,\min} - u_1(t) \\ u_2(t) - u_{2,\max} \\ u_{2,\min} - u_2(t) \end{bmatrix} \leq \mathbf{0}\ \forall t \in [0, T], \tag{4.45}$$

where we have expressed the equality constraint ${}^{\mathrm{F}}x(l(t)) = 0$ by the equivalent of two inequalities with inverted sign. The initial and final state constraint functions, $\mathbf{r}_{\text{init}}$ and $\mathbf{r}_{\text{end}}$ are given in (4.35) and (4.40), respectively, so we have

$$\begin{aligned} \mathbf{r}_{\text{init}}(\tilde{\mathbf{s}}(0)) &= \mathbf{r}_{\text{init}}(\mathbf{s}(0)) = \mathbf{0}, \\ \mathbf{r}_{\text{end}}(\tilde{\mathbf{s}}(T)) &= \mathbf{r}_{\text{end}}(\mathbf{s}(T)) \leq \mathbf{0}. \end{aligned} \tag{4.46}$$

For the acceleration criticality (4.2), i.e., $\mathrm{c}_t = \frac{1}{\mu g}\,\|\mathbf{a}(t)\|$, we can now compactly

state the problem of finding a label $c = \mathrm{c}(\mathcal{S})$ for a driving scene $\mathcal{S}$ as,

MinMax Acceleration OCP

$$
\begin{aligned}
\mathrm{c}(\mathcal{S}) &= \sqrt{z_{\text{opt}}}, \\
z_{\text{opt}} = \min_{\substack{z,\tilde{\mathbf{s}}(t),\mathbf{u}(t) \\ \forall t \in [0,T]}} & \; z, \text{ s.t.} \\
\frac{1}{\mu^2 g^2} \left\| \mathbf{a}(t) \right\|^2 &\leq z, \\
\dot{\tilde{\mathbf{s}}}(t) - \mathrm{f}(\tilde{\mathbf{s}}(t), \mathbf{u}(t)) &= \mathbf{0}, \\
\mathbf{r}_{\text{init}}(\tilde{\mathbf{s}}(0)) &= \mathbf{0}, \\
\mathbf{r}_{\text{end}}(\tilde{\mathbf{s}}(T)) &\leq \mathbf{0}, \\
\mathbf{g}_{\text{path}}(\tilde{\mathbf{s}}(t), \mathbf{u}(t)) &\leq \mathbf{0} \; \forall t \in [0, T],
\end{aligned} \tag{4.47}
$$

where the functions $\mathbf{r}_{\text{init}}$, $\mathbf{r}_{\text{end}}$, and $\mathbf{g}_{\text{path}}$ are all parameterized by the entries of the scene tuple $\mathcal{S}$.

4.5.1 Infeasibility and Interpretability

When the geometry and kinematics of a driving situation are such that a collision is unavoidable, the optimal control problem 4.47 becomes infeasible, i.e., it is not possible to compute a collision avoidance trajectory that satisfies the ODE-, anti-collision-, and lane keeping constraints. Here, we describe the consequences of infeasibility. We also discuss alternative OCP formulations that trade-off between infeasibility, model detail, and interpretability.

For an application in criticality regression, the infeasibility reduces the amount of training and test data we can generate. For a criticality classifier that has to discriminate between *Fire* and *No-Fire* decisions, the inability to solve $\mathrm{c}(\mathcal{S})$ could be interpreted as a *Fire* label since a high criticality correlates with the inability to find a collision avoidance trajectory. However, it is important to distinguish the kinds of conclusions we can draw from successful or failed optimization attempts, i.e.,

- finding a feasible solution proves the feasibility of the OCP,
- not finding a feasible solution does not prove that the OCP is infeasible.

Typically, proving the infeasibility of a non-convex optimization problem is not possible as it would require checking the feasibility of all possible state and control functions. Consequently, it is possible that a driving scenario $\mathcal{S}$ would allow a feasible avoidance trajectory, but the solver reports the problem as infeasible. In that case, we cannot compute a reference criticality label $\mathrm{c}(\mathcal{S})$ for the driving scenario $\mathcal{S}$. It is possible to obtain more feasible solutions for varying driving scenes by considering

alternative formulations for the criticality OCP. In the formulation of an OCP for criticality labeling, we have two main degrees of freedom to influence the feasibility of the OCP, i.e., the representation of the actor limitations and the choice of the tire model. In the problem formulation (4.47), we minimize the peak acceleration of the avoidance trajectory of the host vehicle. The actor limitations are incorporated through upper and lower bounds on the rates of change of the tangential tire force and steering angle inputs u_1, u_2.

An alternative problem formulation consists of minimizing the constraint violation of the inputs u_1, u_2 without strictly limiting them to an upper or lower bound. Alternatively, the control constraints can be satisfied by minimizing the maximum constraint violation. This approach is well known in the robotics literature and has been used in the context of active safety in, e.g., [47], [114], [50]. Observe that the control constraints of the criticality OCP (see, Equation (4.45)) can be rewritten as

$$\begin{bmatrix} u_1 - u_{1,\max} \\ u_{1,\min} - u_1 \\ u_2 - u_{2,\max} \\ u_{2,\min} - u_2 \end{bmatrix} \leq \mathbf{0} \Leftrightarrow \begin{bmatrix} \frac{u_1}{u_{1,\max}} \\ \frac{u_1}{u_{1,\min}} \\ \frac{u_2}{u_{2,\max}} \\ \frac{u_2}{u_{2,\min}} \end{bmatrix} \leq \mathbf{1}, \tag{4.48}$$

where we use the fact that $u_{1,\min}, u_{2,\min} < 0$ and $u_{1,\max}, u_{2,\max} > 0$. To move the input constraints into the cost function, we rewrite these constraints in scalar form, i.e.,

$$\begin{bmatrix} \frac{u_1}{u_{1,\max}} \\ \frac{u_1}{u_{1,\min}} \\ \frac{u_2}{u_{2,\max}} \\ \frac{u_2}{u_{2,\min}} \end{bmatrix} \leq \mathbf{1} \Leftrightarrow \max \left\{ \frac{u_2(t)}{u_{2,\max}}, \frac{u_2(t)}{u_{2,\min}}, \frac{u_1(t)}{u_{1,\max}}, \frac{u_1(t)}{u_{1,\min}} \right\} \leq 1. \tag{4.49}$$

We can now define an alternative cost function for the collision avoidance OCP,

$$\begin{aligned} \mathrm{c}'_t(\mathbf{s}, \mathbf{u}) &= \max \left\{ \frac{\|\mathbf{a}(t)\|_2}{\mu g}, \frac{u_2(t)}{u_{2,\max}}, \frac{u_2(t)}{u_{2,\min}}, \frac{u_1(t)}{u_{1,\max}}, \frac{u_1(t)}{u_{1,\min}} \right\} \\ \mathrm{c}'_{[0,T]}(\mathbf{s}, \mathbf{u}) &= \max_{t \in [0,T]} \mathrm{c}'_t(\mathbf{s}(t), \mathbf{u}(t)). \end{aligned} \tag{4.50}$$

While the constraint minimization cost function (4.50) is more complex than the max-norm acceleration cost function (4.3), we can still interpret it can as a criticality. Since all components of (4.50) are normalized by their upper and lower bounds, we can think of the optimized value of the cost function as the fraction of the control-capacity of the vehicle that is needed to drive along a trajectory $\mathbf{s}, \mathbf{u}$.

As in the OCP (4.47), we optimize an upper-bound of the squared elements of the criticality function to obtain smooth gradients. The modified optimal control problem

becomes

Constraint Minimization OCP

$$
\begin{aligned}
\mathrm{c}(\mathcal{S}) &= \sqrt{z_{\text{opt}}}, \\
z_{\text{opt}} &= \min_{\substack{z,\tilde{\mathbf{s}}(t),\mathbf{u}(t) \\ \forall t\in[0,T]}} z, \text{ s.t.} \\
\frac{1}{\mu^2 g^2}\left\|\mathbf{a}(t)\right\|^2 &\leq z, \\
\frac{u_1^2}{u_{1,\min}^2} &\leq z, \\
\frac{u_2^2}{u_{2,\min}^2} &\leq z, \\
\dot{\tilde{\mathbf{s}}}(t) - \mathrm{f}(\tilde{\mathbf{s}}(t),\mathbf{u}(t)) &= \mathbf{0}, \\
\mathbf{r}_{\text{init}}(\tilde{\mathbf{s}}(0)) &= \mathbf{0} \\
\mathbf{r}_{\text{end}}(\tilde{\mathbf{s}}(T)) &\leq \mathbf{0} \\
\mathbf{g}'_{\text{path}}(\tilde{\mathbf{s}}(t),\mathbf{u}(t)) &\leq \mathbf{0}\ \forall t\in[0,T].
\end{aligned}
\tag{4.51}
$$

Compared to the original path constraint function (4.45) of the OCP (4.47), the path constraint function $\mathbf{g}'_{\text{path}}$ does not include any control constraints, since the control constraints are now part of the optimization function. The new path constraint is

$$
\mathbf{g}'_{\text{path}}(\tilde{\mathbf{s}}(t),\mathbf{u}(t)) = \begin{bmatrix}
{}^{\mathrm{F}}x(l(t)) \\
-{}^{\mathrm{F}}x(l(t)) \\
{}^{\mathrm{F}}y(l(t)) - B_{\mathrm{l}} + B_{\Delta} \\
-B_{\mathrm{r}} + B_{\Delta} - {}^{\mathrm{F}}y(l(t)) \\
{}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{h},1} - {}^{\mathrm{I}}b(t) \\
\vdots \\
{}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{h},N_{\mathrm{h}}} - {}^{\mathrm{I}}b(t) \\
{}^{\mathrm{I}}b(t) - {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{obs},1} \\
\vdots \\
{}^{\mathrm{I}}b(t) - {}^{\mathrm{I}}\mathbf{n}_{\phi}^{\mathrm{T}}(t)\,{}^{\mathrm{I}}\mathbf{c}_{\mathrm{obs},N_{\text{obs}}}
\end{bmatrix} \leq \mathbf{0}\ \forall t\in[0,T]. \tag{4.52}
$$

Since the box-constraints on the steering control u_2 are symmetric, i.e., $u_{1,\max} = -u_{1,\min}$ (see Table 4.2), the constraint on the squared value of u_2 is equivalent to minimizing the terms $\frac{u_2}{u_{2,\min}}$ and $\frac{u_2}{u_{2,\max}}$ of the criticality (4.50). The box-constraints on the force rate control u_1 are not symmetric, i.e., $u_{1,\min} \neq u_{1,\max}$, see Table 4.2. However, in practice, collision avoidance doesn't require intense acceleration, so the upper bound $u_1 \leq u_{1,\max}$ is never tight and can be dropped. It is possible to optimize unsymmetric box-constraints by optimizing the linear control elements of

Table 4.1: Variants of the criticality labeling OCP

Shorthand	Optimal Control Problem	Tire Model
MINA-NLT	MinMax Acceleration (4.47) $\max_{t\in[0,T]} \frac{\|\mathbf{a}(t)\|}{\mu g}$	Magic Formula & Force Coupling (2.52)
MINA-LT	MinMax Acceleration (4.47) $\max_{t\in[0,T]} \frac{\|\mathbf{a}(t)\|}{\mu g}$	Linear Side Forces (2.47)
MINDYN-LT	Constraint Minimization (4.51) $\max_{t\in[0,T]} \max\left\{ \frac{\|\mathbf{a}(t)\|_2}{\mu g}, \frac{u_2(t)}{u_{2,\max}}, \frac{u_2(t)}{u_{2,\min}}, \dots \right\}$	Linear Side Forces (2.47)

(4.50). However, this would have the drawback of changing the trade-off between acceleration and control constraints, by only squaring the acceleration norm.

By moving the control constraint of (4.47) into the cost function, the OCP (4.51) has strictly more feasible trajectories thus increasing the availability of criticality labels. Furthermore, all solutions of (4.51) with an optimal value smaller than one are feasible solutions of (4.47).

4.5.2 Considered OCP Variants and Parameters

We consider three variants of the labeling OCP, listed in Table 4.1 which differ in their formulation and tire model. The first two OCPs minimize the maximal 2-norm of the acceleration vector as formulated in (4.47). They differ in the modeling detail of the tire forces. MINA-NLT uses the force-coupled tire model (2.52) which incorporates the nonlinear saturation of the side force as well as the limitation of the tire force vector norm. This force saturation puts a limit on the maneuverability of the host vehicle in addition to the box constraints on the steering and force rate inputs $u_2,\ u_1$. Due to this model detail, MINA-NLT is the most realistic dynamic model considered here. However, since it also incorporates the most constraints on the vehicle dynamics, the set of driving situations $\mathcal{S}$ that can be labeled with a criticality value using MINA-NLT is the smallest among the considered OCP formulations.

MINA-LT differs from MINA-NLT by its use of the linear tire model in (2.47). Thus MINA-LT is only constrained by the input limits and should lead to increased feasibility, albeit at the cost of realism of the simulation.

We abbreviate the constraint minimization OCP (4.51) as MINDYN-LT. By using the least number of constraints, its availability is the largest of the considered OCPs.

Vehicle and OCP parameters In Table 4.2, we list all parameters used for the host vehicle as well as constraint and margin values used in the OCP formulations. The vehicle parameters approximately represent an SUV in the category of an Audi Q7. We have discussed the state and control constraint values in the preceding sections. Their

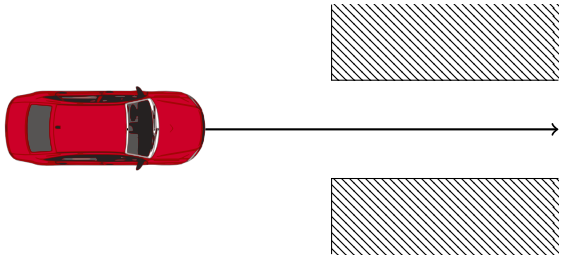


Figure 4.3: The passage scenario. Only a straight trajectory can avoid a collision.

values depend on the performance characteristics of the vehicle. While the chosen parameters are well suited to illustrate and study the criticality labeling problem, in practice, all parameters should be carefully adjusted to the considered vehicle.

4.6 A Counterexample and some Extensions

The minimal peak acceleration of an evasion trajectory is an intuitive and physically-motivated measure of the criticality of a rear-end collision scenario. However, there are scenarios where the acceleration or other state variables do not accurately reflect the threat of the driving situation. Consider, e.g., the scenario[6] depicted in Figure 4.3. The EGO vehicle is driving towards a small passage which is as wide as the vehicle itself. This is a situation, where the peak avoidance acceleration does not reasonably reflect the threat. The avoidance acceleration is zero while the scenario is highly threatening due to the tightness of the passage.

In this situation, what is missing to represent the threat is a measure of the robustness of the avoidance trajectory. The passage is threatening because a slight steering motion or an imperfect estimate of the initial state lead to a collision. This observation also holds for situations that require steering maneuvers: if there exists only a single avoidance trajectory, and any deviation from that trajectory leads to a collision, the situation is threatening.

A first solution to handle the passage scenario is to fuse obstacles that are close together into a single, enlarged obstacle during environment perception. However, the decision on when to fuse obstacles can turn out to be a delicate algorithmic problem in itself. Another approach to better represent the criticality is to compute the sensitivity of the criticality measure to perturbations of the initial state of the EGO vehicle. This sensitivity is studied in [50, Ch. 4] for variable-time avoidance trajectories. Moreover, the criticality could be derived from reachable sets. A reachable set is the set of states that can be reached from the initial state in certain time. The larger the reachable set and the set of feasible trajectories, the more likely a collision can be avoided. Reachable sets for collision avoidance have been studied, e.g., in [37] and [110].

[6] This scenario was proposed by Dr.rer.nat. Frank Keck during a status meeting.

Table 4.2: Parameters of the criticality optimal control problem

parameter	symbol	value
mass	m	2070kg
moment of inertia	I	2750kgm^2
host vehicle body length	L	5.05m
host vehicle body width	W	2.2m
COG distance from front wheel	l_f	1.3m
COG distance from rear wheel	l_r	1.45m
tire model - stiffness factor	B	$0.2391/^\circ$
tire model - shape factor	C	1.19
tire model - peak factor	D	3750N
tire model - curvature factor	E	-0.678
lane keeping margin	B_Δ	$0.5W$
prediction horizon	T	2.5s
yaw tolerance at $t = T$	ψ_{end}	10°
min. velocity	v_{min}	$1\ \text{m/s}$
max. steer angle	δ_{max}	∞
min. steer angle	δ_{min}	$-\infty$
max. steer rate	$u_{2,\text{max}}$	$\frac{360}{15}^\circ/\text{s}$
min. steer rate	$u_{2,\text{min}}$	$-\frac{360}{15}^\circ/\text{s}$
friction coefficient	μ	1.01
min. tire force	$F_{\text{t,min}}$	$-\mu mg$
max. tire force	$F_{\text{t,max}}$	∞
min. tire force rate	$u_{1,\text{min}}$	$-\frac{\mu mg}{0.2}$
max. tire force rate	$u_{1,\text{max}}$	$-5u_{1,\text{min}}$

Chapter 5

Numerical Criticality Labeling and Simulation Results

To solve the criticality OCP, we employ the Direct Multiple Shooting [44] technique as detailed in Chapter 3. To this end, we need to numerically integrate the ordinary differential equation of the vehicle model and formulate an initial guess for the optimization variables.

5.1 Multiple Shooting and Nonlinear Optimization Problem

The prediction horizon $[0, T]$ is divided into N intervals of equal length Δt, during which the control inputs $\mathbf{u}[n]$ are kept constant. On each interval, we solve the ordinary differential equation of the host vehicle dynamics (2.39) using numerical integration, i.e.,

$$\hat{\mathbf{s}}[n+1] = \mathrm{F}(\Delta t; \mathbf{s}[n], \mathbf{u}[n]), \tag{5.1}$$

where $\mathrm{F}(\Delta t; \mathbf{s}[n], \mathbf{u}[n])$ denotes the numerical integration over the interval Δt given the initial value $\mathbf{s}[n]$ and control input $\mathbf{u}[n]$.

In our implementation, we use M steps of fourth-order Runge-Kutta [115] (RK4) integration. This means that we partition the interval $[t_n, t_{n+1}]$ into M subintervals of

equal length h. On each subinterval we perform $M = 10$ Runge-Kutta steps, i.e.,

$$\begin{aligned} \mathrm{F}_{\mathrm{RK4}}(h; \mathbf{s}, \mathbf{u}) &= \frac{h}{6}(k_1 + 2k_2 + 2k_3 + k_4), \\ k_1 &= \mathrm{f}(\mathbf{s}, \mathbf{u}), \\ k_2 &= \mathrm{f}(\mathbf{s} + \frac{h}{2}k_1, \mathbf{u}), \\ k_3 &= \mathrm{f}(\mathbf{s} + \frac{h}{2}k_2, \mathbf{u}), \\ k_4 &= \mathrm{f}(\mathbf{s} + hk_3, \mathbf{u}), \end{aligned} \tag{5.2}$$

with the ODE $\dot{\mathbf{s}} = \mathrm{f}(\mathbf{s}, \mathbf{u})$ given in (2.39). Each successive RK4 step with index m uses the result of the preceding integration as its initial value, i.e.,

$$\mathrm{F}_{m,n}(h) = \begin{cases} \mathbf{s}[n] & \text{if } m = 0 \\ \mathrm{F}_{\mathrm{RK4}}(h; \mathrm{F}_{m-1,n}(h), \mathbf{u}[n]) & \text{if } m = 1, \ldots, M \end{cases} \tag{5.3}$$

The numerical integrator F in (5.1) thus given by

$$\mathrm{F}(\Delta t; \mathbf{s}[n], \mathbf{u}[n]) = \mathrm{F}_{M,n}(h),\ h = \frac{\Delta t}{M}. \tag{5.4}$$

In order to find a continuous state trajectory, the numerical solutions of the ODE of one interval must match exactly the initial value of the following interval, i.e.,

$$\mathbf{s}[n+1] = \hat{\mathbf{s}}[n+1], \tag{5.5}$$

which leads to so-called multiple-shooting constraints as discussed in Chapter 3,

$$\mathbf{s}[n+1] - \mathrm{F}(\Delta t; \mathbf{s}[n], \mathbf{u}[n]) = \mathbf{0}\ \forall n = 0, \ldots, N-1. \tag{5.6}$$

We can now write the discretized approximation of (4.47) as

$$\begin{aligned} \mathrm{c}(\mathcal{S}) &= \sqrt{z_{\mathrm{opt}}} \\ z_{\mathrm{opt}} = \min_{\substack{z \\ \tilde{\mathbf{s}}[0],\ldots,\tilde{\mathbf{s}}[N] \\ \mathbf{u}[0],\ldots,\mathbf{u}[N-1]}} & z,\ \text{s.t.} \\ \frac{1}{\mu^2 g^2} \left\|\mathbf{a}[n]\right\|^2 &\leq z,\ \forall n = 0, \ldots, N, \\ \mathbf{r}_{\mathrm{init}}(\tilde{\mathbf{s}}[0]) &= \mathbf{0}, \\ \mathbf{r}_{\mathrm{end}}(\tilde{\mathbf{s}}[N]) &\leq \mathbf{0}, \\ \mathbf{s}[n+1] - \mathrm{F}(\Delta t; \mathbf{s}[n], \mathbf{u}[n]) &= \mathbf{0}, \\ \mathbf{g}_{\mathrm{path}}(\tilde{\mathbf{s}}[n], \mathbf{u}[n]) &\leq \mathbf{0},\ \forall n = 0, \ldots, N-1. \end{aligned} \tag{5.7}$$

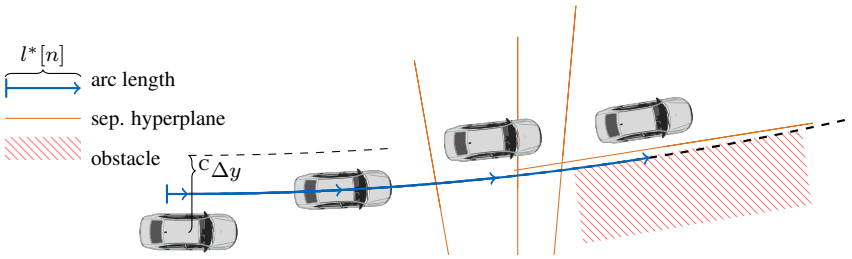


Figure 5.1: Initialization of the optimization variables for the numerical solution of the OCP (5.7)

The discretization of the constraint minimization problem (4.51) reads as

$$
\begin{aligned}
\mathrm{c}(\mathcal{S}) &= \sqrt{z_{\text{opt}}} \\
z_{\text{opt}} = \min_{\substack{z \\ \tilde{\mathbf{s}}[0],\ldots,\tilde{\mathbf{s}}[N] \\ \mathbf{u}[0],\ldots,\mathbf{u}[N-1]}} \quad & z, \text{ s.t.} \\
\frac{1}{\mu^2 g^2} \left\| \mathbf{a}[n] \right\|^2 &\leq z, \ \forall n = 0, \ldots, N, \\
\frac{u_1^2[n]}{u_{1,\min}^2} &\leq z, \\
\frac{u_2^2[n]}{u_{2,\min}^2} &\leq z, \\
\mathbf{r}_{\text{init}}(\tilde{\mathbf{s}}[0]) &= \mathbf{0}, \\
\mathbf{r}_{\text{end}}(\tilde{\mathbf{s}}[N]) &\leq \mathbf{0}, \\
\mathbf{s}[n+1] - \mathrm{F}(\Delta t; \mathbf{s}[n], \mathbf{u}[n]) &= \mathbf{0}, \\
\mathbf{g}'_{\text{path}}(\tilde{\mathbf{s}}[n], \mathbf{u}[n]) &\leq \mathbf{0}, \ \forall n = 0, \ldots, N-1.
\end{aligned} \tag{5.8}
$$

These nonlinear optimization problems can be solved using a sparse nonlinear programming solver; we use the CasADI [94] library together with the IPOPT [95] solver for finding locally optimal solutions to the discretized OCPs.

5.2 Initial Guess for the Optimization Variables

An important advantage of the direct multiple shooting method of optimal control is that it is possible to formulate an initial guess for the optimization problem in terms of the state variables $\mathbf{s}[n],\ n = 0, \ldots, N$. The goal of the initialization is to find initial values for the configuration of the host vehicle, i.e., the position ${}^{\mathrm{I}}x[n],\ {}^{\mathrm{I}}y[n]$ and yaw

angle ${}^{\mathrm{I}}\psi$ of the host vehicle. The construction of our initial guess is illustrated in Figure 5.1. As a simplification, we assume that the host vehicle velocity remains constant throughout the prediction interval,

$$v_{\mathrm{h}}^{*}[n] = v_{\mathrm{h},0},\ n = 0, \ldots, N, \tag{5.9}$$

with the initial velocity v_0 given by the scene tuple $\mathcal{S}$.

We specify the initial guess for the position of the vehicle in terms of the curvilinear coordinates $l[n]$ and ${}^{\mathrm{C}}y[n]$. We assume that the vehicle drives parallel to the lane center given by the path ${}^{\mathrm{I}}\mathbf{r}(l)$. Thus, we can initialize the arc length variable l as

$$l^{*}[n] = l_0 + n\Delta t v_{\mathrm{h},0},\ n = 0, \ldots, N. \tag{5.10}$$

Equation 5.10 contains a further simplification by setting the arc increment per time step on the lane center equal to the arc length increment of the host vehicle COG. It would be possible to do better and adjust the arc increment to the local curvature of the tube, however, in practice, we did not observe any problems with this simplification. Next, we initialize the normal coordinate ${}^{\mathrm{C}}y$ of the host vehicle in curvilinear coordinates. To obtain a trajectory that is collision-free, the host vehicle must be displaced laterally by a total distance of ${}^{\mathrm{C}}\Delta y$ as illustrated in Figure 5.1. Increasing this lateral displacement with every time step, the initialization writes as

$$\begin{aligned} {}^{\mathrm{C}}\Delta y &= |\,{}^{\mathrm{C}}y_{\mathrm{obs},0} - {}^{\mathrm{C}}y_0| + 0.5W_{\mathrm{h}}, \\ {}^{\mathrm{C}}y^{*}[n] &= {}^{\mathrm{C}}y_0 + w\,{}^{\mathrm{C}}\Delta y, \end{aligned} \tag{5.11}$$

where ${}^{\mathrm{C}}y_0$ is the initial offset coordinate of the host vehicle in curvilinear coordinates given by the scene tuple $\mathcal{S}$, ref. (2.12). The weight factor w in (5.11) is given by

$$w = \mathrm{sign}({}^{\mathrm{C}}y_0)\frac{n}{n_{\max}}, \tag{5.12}$$

which leads to a linear increase of the lateral offset of the host vehicle. In our simulation we have used $n_{\max} = 7$ for $N = 30$ and $T = 2.5$, which was determined as a suitable choice by trial and error. Since the viability of this choice for $n_{\max}$ depends on the driving situation, its value can be adapted based on the initial time-to-collision t_{ttc} of the driving scene, i.e.,

$$n_{\max} = \left\lfloor \frac{t_{\mathrm{ttc}}}{\Delta t} \right\rfloor, \tag{5.13}$$

where the time-to-collision t_{ttc} is given in (6.41) and $\lfloor\cdot\rfloor$ rounds down to integers. This choice would ensure that the lateral displacement ${}^{\mathrm{C}}\Delta y$ for collision avoidance is reached before the host vehicle geometry advances past the obstacle position.

The position of the host vehicle in road fixed coordinates is obtained by transforming the road aligned coordinates to Cartesian coordinates, i.e.,

$$\begin{bmatrix} {}^{\mathrm{I}}x^* \\ {}^{\mathrm{I}}y^* \end{bmatrix}[n] = {}^{\mathrm{I}}\mathbf{r}(l^*[n]) + {}^{\mathrm{C}}y[n]\,{}^{\mathrm{I}}\mathbf{n}(l^*[n]), \tag{5.14}$$

where the position of the curvilinear arc ${}^{\mathrm{I}}\mathbf{r}$ is given in (2.2) and the normal vector ${}^{\mathrm{I}}\mathbf{n}(l[n])$ is defined in (2.10).

Since we assume that the host vehicle drives parallel to the curvilinear lane center, we initialize the yaw angle using the local tangent angle

$${}^{\mathrm{I}}\psi^*[n] = {}^{\mathrm{I}}\tau(l^*[n]), \tag{5.15}$$

where ${}^{\mathrm{I}}\tau$ is given in (2.3).

At this point, we have formulated initial guesses for the position, yaw angle, and velocity of the host vehicle for all time steps $n = 0, \dots, N$ as well as the arc length variable l for lane keeping. We initialize all other state and control variables as zero giving us the initial guess for the vehicle state vector and controls,

$$\begin{aligned} \mathbf{s}^*[n] &= \begin{bmatrix} {}^{\mathrm{I}}x^*[n] & {}^{\mathrm{I}}y^*[n] & v_{\mathrm{h}}^*[n] & 0 & {}^{\mathrm{I}}\psi^*[n] & 0 & 0 & 0 \end{bmatrix}^{\mathrm{T}}, \\ \mathbf{u}^*[n] &= \begin{bmatrix} 0 & 0 \end{bmatrix}^{\mathrm{T}} \ \forall n, \\ z^* &= 100. \end{aligned} \tag{5.16}$$

Initialization of the separating hyperplane variables Next, we need to find an initialization for the auxiliary variables for collision avoidance ${}^{\mathrm{I}}b[n]$ and ${}^{\mathrm{I}}\phi[n]$ to satisfy the anti-collision constraints (4.18). In our experiments, we have observed that the initialization of the separating hyperplane is critical for the ability to converge to a feasible solution of the OCP. In fact, we have not observed any optimization that has successfully converged when the initialization did not satisfy the anti-collision constraints for all time steps.

In the following, we assume that we have chosen an initialization for the position and yaw angle where the geometries do not intersect. As illustrated in Figure 5.2, we construct the hyperplane such that it is perpendicular to the shortest line between the host and obstacle bodies. This amounts to finding points inside the host vehicle and obstacle polyhedrons $\mathcal{B}_{\mathrm{h}}[n],\ \mathcal{B}_{\mathrm{obs}}[n]$ with minimal Euclidean distance. Formally, we solve the following optimization problem for each time step n:

$$\mathbf{p}_{\mathrm{h}}^*[n],\ \mathbf{p}_{\mathrm{obs}}^*[n] = \underset{\substack{\mathbf{p}_{\mathrm{h}} \in \mathcal{B}_{\mathrm{h}}[n] \\ \mathbf{p}_{\mathrm{obs}} \in \mathcal{B}_{\mathrm{obs}}[n]}}{\arg\min}\ \left\| \mathbf{p}_{\mathrm{h}} - \mathbf{p}_{\mathrm{obs}} \right\|^2. \tag{5.17}$$

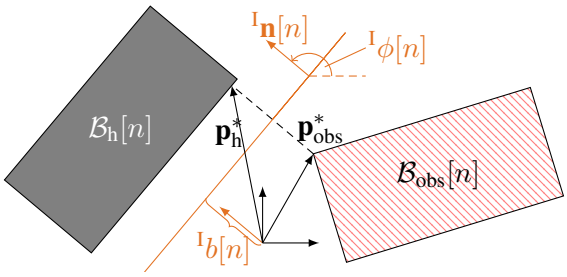


Figure 5.2: Separating hyperplane initialization

This is a convex quadratic optimization problem which is easily solved using off-the-shelve optimization routines. Given the minimum distance points $\mathbf{p}^*_{\mathrm{h}}[n]$ and $\mathbf{p}^*_{\mathrm{obs}}[n]$, a separating hyperplane defined by its offset $^{\mathrm{I}}b[n]$ and the angle of its normal vector $^{\mathrm{I}}\phi[n]$ (ref. (4.17)) is easily computed as perpendicular to the line connecting $\mathbf{p}^*_{\mathrm{h}}$ and $\mathbf{p}^*_{\mathrm{obs}}[n]$. We can compute the angle of the normal vector as

$$^{\mathrm{I}}\phi[n] = \arctan\left(\frac{\Delta\mathbf{p}_y}{\Delta\mathbf{p}_x}\right), \quad \Delta\mathbf{p} = \mathbf{p}^*_{\mathrm{h}} - \mathbf{p}^*_{\mathrm{obs}}[n], \tag{5.18}$$

where $\Delta\mathbf{p}_x, \Delta\mathbf{p}_y$ are the x and y components, respectively, of the vector $\Delta\mathbf{p}$. To compute the hyperplane offset $^{\mathrm{I}}b[n]$, we need to project the point $\mathbf{p} = 0.5(\mathbf{p}^*_{\mathrm{h}} + \mathbf{p}^*_{\mathrm{obs}}[n])$ of the hyperplane onto the normal vector, i.e.,

$$^{\mathrm{I}}b[n] = {}^{\mathrm{I}}\mathbf{n}^{\mathrm{T}}_{\phi}[n]\mathbf{p}, \quad \mathbf{p} = 0.5(\mathbf{p}^*_{\mathrm{h}} + \mathbf{p}^*_{\mathrm{obs}}[n]). \tag{5.19}$$

where the normal vector $^{\mathrm{I}}\mathbf{n}_{\phi}$ is defined in (4.17). It is possible to come up with many alternative construction techniques for the separating hyperplane and we have experimented with a few other heuristics. The proposed construction technique for the separating hyperplane has the useful property that it does not introduce any additional tuning parameters.

In order to further understand the utility of using the multiple-shooting method over the single shooting method, it is instructive to consider how an initialization for a single-shooting formulation would have to be derived. In a single-shooting formulation, the state vectors $\mathbf{s}[n]$ are not variables of the nonlinear program. Instead, they are computed from the initial state $\mathbf{s}_0$ and the control variables $\mathbf{u}[n]$ by integrating over the interval $[0, t_n]$, i.e.,

$$\hat{\mathbf{s}}[n] = \mathrm{F}(n\Delta t; \mathbf{s}_0, \mathbf{u}[0], \ldots, \mathbf{u}[n-1]). \tag{5.20}$$

Therefore, an initial guess for the solution of the single shooting formulation must be formulated in terms of the control variables $\mathbf{u}[n]$, i.e., in terms of the rate of change of the tangential tire force $F_{\mathrm{t}}[n]$ and the rate of change of the steering angle $\delta[n]$. Evidently, such an initialization would be more cumbersome to formulate than the initialization derived in this Chapter which relies on intuitive geometry and kinematics.

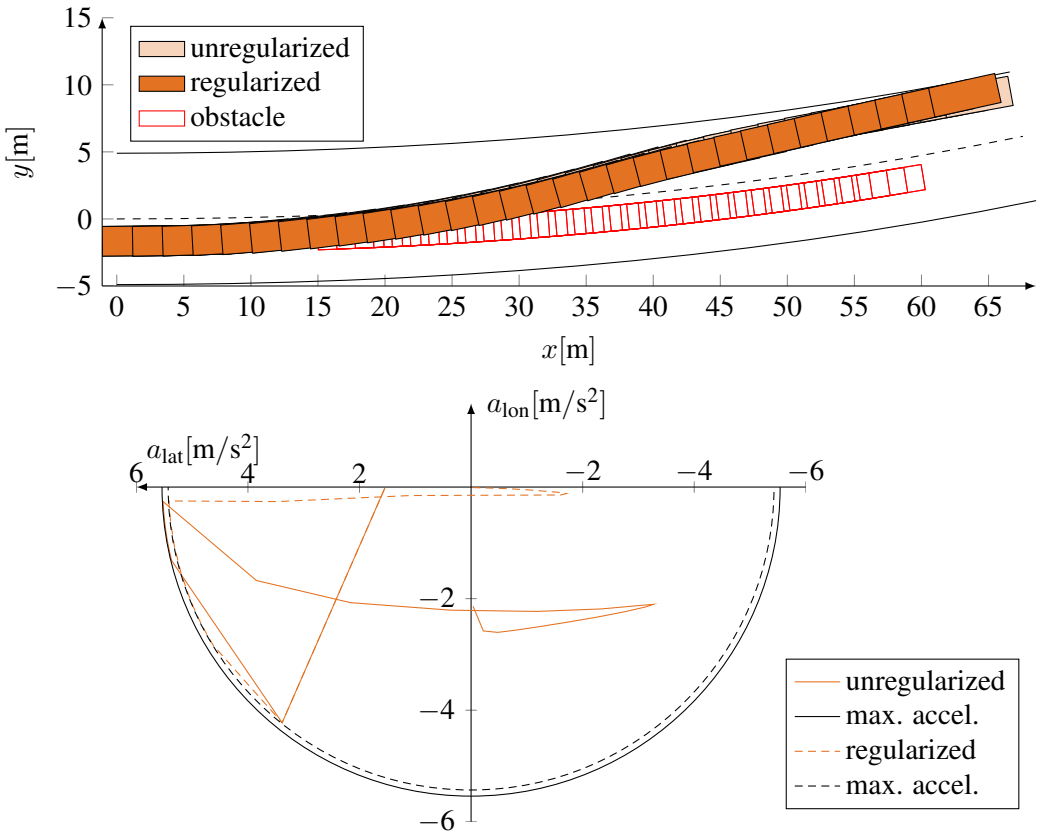


Figure 5.3: Effect of regularization of the cost function on an evasion maneuver on a curved lane. Top: top-down view of the evasion maneuver. Bottom: acceleration over time. Scene tuple: $v_{\text{h}} = 27\text{m/s}$, ${}^{\text{C}}y = -1.66\text{m}$, $\Delta x_0 = 14.9\text{m/s}$, $v_{\text{obs}} = 16.5\text{m/s}$, $a_{\text{obs}} = -0.3\text{m/s}^2$, $B_{\text{l}} = B_{\text{r}} = 4.9\text{m}$, $1/c_0 = 475\text{m}$, $\kappa = 2.54 \cdot 10^{-5} 1/\text{m}^2$.

5.3 Regularization, Scaling, and Multi-Start

The numerical solution of the criticality OCP (4.47) can lead to solutions that are locally optimal with respect to the cost function but that show unconvincing trajectories after the initial avoidance maneuver.

To illustrate this phenomenon, Figure 5.3 shows the optimized trajectory for an example scenario, once with and once without regularization, using the OCP (5.7). Without regularization, the part of the trajectory that looks suspect are the states after the initial collision avoidance maneuver (turning left): even though the collision is already avoided, the host vehicle keeps decelerating as can be seen from the acceleration plot. This seems unmotivated, since there is nothing in our OCP formulation that encourages the host vehicle trajectory to try and reach a low-velocity state.

To understand this behavior, note that the vehicle states after the initial collision avoidance do not influence the optimized cost of the OCP (5.7) since the host vehicle does not have to avoid another collision. The impending collision is avoided during the initial time steps $n \leq 15$ where the peak of the acceleration norm is attained. Since the peak of the acceleration norm does not depend on the control inputs for $n > 15$, the optimized trajectory can take any shape for $n > 15$ as long as it satisfies all constraints.

Regularization To obtain trajectories that tend towards low acceleration norms after the collision has been avoided, we add a regularization component to the cost function of the OCPs (4.47) and (4.51), i.e.,

$$\min z \Rightarrow \min \left(z + \lambda \frac{1}{N+1} \sum_{n=0}^{N} \frac{\|\mathbf{a}[n]\|^2}{\mu^2 g^2} \right), \tag{5.21}$$

where the longitudinal and lateral components of the acceleration vector $\mathbf{a}[n]$ are functions of the state and control vectors $\mathbf{s}[n],\ \mathbf{u}[n]$ as defined in (2.25) and (2.26). The regularization factor $\lambda > 0$ is used to trade the trajectory simplicity against the upper bound z of the acceleration norm. This regularization term penalizes the average acceleration norm encouraging simple trajectories in the sense that steady-state driving is penalized less than permanent braking and steering.

The regularization term does not significantly change the optimal value of the OCP. Since the maximum is greater than the average acceleration, and since we constrain the acceleration by $\frac{\|\mathbf{a}[n]\|^2}{\mu^2 g^2} \leq z \ \forall n$ in the criticality OCP (5.7) and (5.8), the relative error due to the regularization term is bounded by

$$e^2 = \frac{\lambda \frac{1}{N+1} \sum_{n=0}^{N} \frac{\|\mathbf{a}[n]\|^2}{\mu^2 g^2}}{z} \overset{\max_n \frac{\|\mathbf{a}[n]\|^2}{\mu^2 g^2} \leq z}{\leq} \lambda \underbrace{\frac{\frac{1}{N+1} \sum_{n=0}^{N} \|\mathbf{a}[n]\|^2}{\max_n \|\mathbf{a}[n]\|^2}}_{\leq 1} \leq \lambda, \tag{5.22}$$

where we also assume that the friction coefficient μ is smaller or equal to one. In our simulations we use a regularization factor of $\lambda = 0.001$. As the criticality is the square root of z_{opt}, this choice of λ makes the relative error e of the optimal criticality due to regularization smaller than 3.2 percent. While the regularization does not significantly influence the optimal value of the OCP it contributes to finding less complex evasion trajectories than the unconstrained formulation.

The effect of the regularization is visible in Figure 5.3. Again, after the collision with the obstacle is avoided, the host vehicle changes its trajectory from a left turn to a right turn to not veer off the road. Then, the vehicle returns to a steady driving state by reducing the longitudinal and lateral accelerations.

Table 5.1: Solver settings used for IPOPT

Setting Nb.	*mu_strategy*	*obj_scaling_factor*	*nlp_scaling_max_gradient*
1	'monotone'	1	100
2	'monotone'	1	10
3	'adaptive'	1	10
4	'monotone'	0.1	100
5	'adaptive'	0.1	100

Solver Settings and Multi-Start Optimization The convergence speed and the ability to find a feasible solution are noticeably influenced by the scaling of the OCP and by the solution strategy employed in the numerical solver. Therefore, we employ a multi-start approach which uses multiple solution runs of the same driving scene $\mathcal{S}$ using a set of different settings for the IPOPT solver. For five different optimization attempts, we varied the following parameters as listed in Table 5.1,

- *mu_strategy*, which determines the update strategy for the barrier parameter μ used in the Primal-Dual Interior Point algorithm in IPOPT,
- *obj_scaling_factor*, which is a scaling factor applied to the cost function, and
- *nlp_scaling_max_gradient* which is the maximum gradient norm allowed by gradient-based rescaling of the OCP.

The maximum number of iterations *max_iter* was fixed to 600 whereas all other settings were left at their default values. A complete list of the possible solver settings and their default values is available in the IPOPT documentation [116]. When multiple solver settings successfully solved the OCP, the solution with the smallest optimal value was retained. The multi-start procedure is summarized in Algorithm 2. The solver settings in Table 5.1 were obtained via manual trial and error by the author. A more exhaustive approach to the problem of finding optimal parameters for the OCP solver could consist of a systematic parameter search, e.g., using Bayesian Optimization [117] for the solver parameters that maximize the amount of successful solutions on a set of driving scenes.

5.4 Simulation Results and Comparison of OCP Formulations

In this section, we evaluate whether the labels obtained by OCP labeling are *well-behaved* in the sense that the labels are

- *smooth*, meaning that the estimated criticality should not fluctuate strongly for small changes to a driving situation,

Algorithm 2 Multiple start criticality labeling with varying solver settings

Require: scene tuple $\mathcal{S}$, host vehicle parameters
  initialize set of solutions $\mathcal{C} \leftarrow \emptyset$
  compute initial guess using (5.9) – (5.19)
  for $i \in \{1, \ldots, 5\}$ **do**
    solve (5.7) using IPOPT with Setting Nb. i, see Table 5.1
    return *feasible*, optimized $z, \tilde{\mathbf{s}}[0], \ldots, \tilde{\mathbf{s}}[N], \mathbf{u}[0], \ldots, \mathbf{u}[0]$
    if *feasible* $==$ *true* **then**
      $c_i = z$
      $\mathcal{C} \leftarrow \mathcal{C} \cup c_i$
    end if
  end for
  if $\mathcal{C} \neq \emptyset$ **then**
    compute criticality value $c = \min \mathcal{C}$
    return c
  else
    return abort: no solution found
  end if

- *interpretable*, i.e., the threat labels should align with intuition,
- *available*, such that a wide range of driving situations can be labeled.

To compare the different OCP formulations, we consider a set of rear-end collisions with varying initial conditions of the host vehicle speed, the obstacle speed, and the lateral position of the host vehicle on the road. All parameters of the OCPs are listed in Table 4.2. The different formulations are all initialized as described in Section 5.2. In our experiments we have used $N = 30$ control intervals for a prediction horizon of $T = 2.5$s.

Analysis of an example scenario In Figure 5.4, we illustrate the computed avoidance trajectories for a host vehicle velocity of $v_{\mathrm{h}} = 15\mathrm{m/s}$. The lane width is set to $B = 3.5$m and the prediction horizon has a length of $T = 2.5$. The upper half depicts a top-down view of the avoidance trajectories while the lower half shows the traces of the lateral and longitudinal accelerations throughout the prediction horizon. Note that the acceleration for all OCP variants follows a common pattern: starting from zero acceleration, the immediate collision with the obstacle is avoided by a joint braking and steering maneuver. Going from point A to B, the deceleration increases faster than the lateral acceleration. After the initial deceleration phase, the maximum lateral acceleration for the collision avoidance is attained at point B which is almost pure

lateral acceleration. At this point, the collision avoidance constraints are satisfied for the entire maneuver and the vehicle dynamics are primarily influenced by the lane keeping constraint since the host vehicle has a significant yaw angle with respect to the lane direction. To satisfy the lane keeping constraint, the acceleration vector approximately traverses a semi-circle to achieve the maximum lateral acceleration in the opposite direction at point D. Note that the traversal from A to C and from C to D resemble each other. In both segments, the host vehicle velocity is reduced by braking before the maximum lateral acceleration is attained, which is reasonable since the maximum lateral acceleration can be reduced by reducing the host vehicle speed.

In the considered scenario, the use of the nonlinear tire model with force saturation in MINA-NLT does not lead to a significantly different solution compared to the OCP MINA-LT which shares the same cost function. This resemblance of the solutions of MINA-NLT and MINA-LT can be seen even more clearly in Figure 5.5, where the trajectories of the tangential tire force F_t, the steering angle δ and their respective control inputs u_1 and u_2 are depicted. We can see that those states and controls are strongly correlated. One way to understand this lack of impact due to the tire limitations is to recall that both formulations, MINA-NLT and MINA-LT, share the acceleration-norm cost function. This cost function leads to solutions that minimize the peak acceleration which is tightly correlated with the peak tire force. In effect, the optimization will always try to find trajectories that don't bring the tire into force saturation and instead keep it in the linear domain, see Figure 2.4. In the linear domain of the tire, MINA-NLT and MINA-LT are equivalent which explains the observation. Arguably, the biggest difference between the MINA-NLT and MINA-LT is the set of feasible driving scenarios since the tire force saturation is equivalent to a constraint on the tire force vector.

The trajectory obtained by solving the constraint minimization problem MINDYN-LT differs significantly from the trajectories obtained by minimizing the acceleration norm. Most importantly, in the acceleration trace of Figure 5.4, we can observe that the constraint minimization approach leads to an avoidance maneuver with a higher peak acceleration norm. At the same time, the control inputs $u_1,\ u_2$ of the MINDYN-LT solution have smaller peak values compared to MINA-NLT and MINA-LT as can be seen in Figure 5.5. This observation is explained by the structure of the cost function of MINDYN-LT which balances the acceleration norm and the control inputs. Indeed, we can observe that the peak ratios of the acceleration of the acceleration and control inputs are approximately equal in this particular solution, i.e.,

$$\max_{n=0,\dots,N} \frac{\|\mathbf{a}\|[n]}{\mu^2 g^2} \approx \max_n \frac{|u_2[n]|}{u_{2,\min}} \approx \max_n \frac{|u_1[n]|}{u_{1,\min}} = 0.7478, \tag{5.23}$$

which is the criticality value of the simulated driving situation. The fact that these peak values are equal is explained by noting that the peak acceleration and the control inputs

are inversely coupled. In other words, the lateral or longitudinal acceleration peaks can be lower if the accelerations are increased faster by using higher absolute steering and force rates and vice versa. In the solutions of MINA-NLT and MINA-LT depicted in Figure 5.5, this inverse coupling is manifested by the fact that the control inputs jump to their maximum values right at the start of the avoidance maneuver. Conversely, any criticality estimation based on avoidance accelerations that does not take rate constraints into account, noticeably underestimates the peak avoidance accelerations. In addition to rate constraints, actor dead times or the reaction times, which are not considered here, can further increase the peak acceleration values.

Macro behavior of the OCP formulations In order to study the smoothness, interpretability, and feasibility of the OCP formulations, we compute the criticality labels of lateral avoidance trajectories for varying initial target distances, host vehicle speeds, and lateral offsets. This allows us to analyze the effects that these parameters have on the criticality values. Figure 5.6 depicts the criticality values $\mathrm{c}(\mathcal{S})$ obtained by solving MINA-NLT, MINA-LT, and MINDYN-LT on a straight lane with width $B = 3.5\mathrm{m}$.

An important observation is that the obtained criticality values are smooth and monotonic functions of the initial distance, velocity and offset. The criticality increases with the relative velocity and a larger overlap between the host vehicle and decreases with distance which conforms well with intuition. Evidently, the fact that the optimal control problems are non-convex does not stand in the way of obtaining interpretable and smooth labels for complex avoidance maneuvers with braking and steering. There is one simulation where we can observe sub-optimality in the solution for MINDYN-LT with $v_\mathrm{h} = 30\mathrm{m/s}$ and $y = 0\mathrm{m}$ which is depicted in the bottom right subplot of Figure 5.6. At a distance of approximately 22m the criticality value is noticeably elevated compared to the neighboring criticality values. Thus, it might be possible to find a lower criticality value for that particular scenario, e.g., by using multiple different initialization techniques and choosing the best result.

For the scenarios with low relative velocity $v_\mathrm{h} = 15\mathrm{m/s}$ in the first row of Figure 5.6 we can observe almost no numerical difference between the criticality values obtained by the different OCP formulations. As discussed above, the MINDYN-LT OCP has to balance the peak acceleration and controls, which causes slightly higher criticality labels for the scenarios with $v_\mathrm{h} = 30\mathrm{m/s}$. Despite these small differences, all considered solutions yield interpretable criticality labels for the considered scenario.

An important difference between the OCP formulations is the availability of criticality labels. The control-input-constrained OCP MINA-NLT with tire force saturation is unable to provide criticality labels for small obstacle distances. The OCP version with a linear tire model MINA-LT provides the same criticality values for a larger set of avoidance scenarios and is almost always able to provide criticality values above one. Therefore, we can conclude that MINA-LT is preferable over MINA-

NLT for the criticality labeling task. The MINDYN-LT OCP can be solved even for scenarios with a criticality larger than one, i.e., we are able to obtain a criticality value even if the collision is physically unavoidable.

To summarize, all considered OCP variations can provide smooth and monotonous criticality values. The constraint minimization OCP MINDYN-LT has the highest availability at the cost of a slightly worse interpretability of the labels due to a more complex cost function. A good compromise between interpretability and availability is achieved by the optimization of the maximum acceleration with a linear tire model as in the OCP MINA-LT.

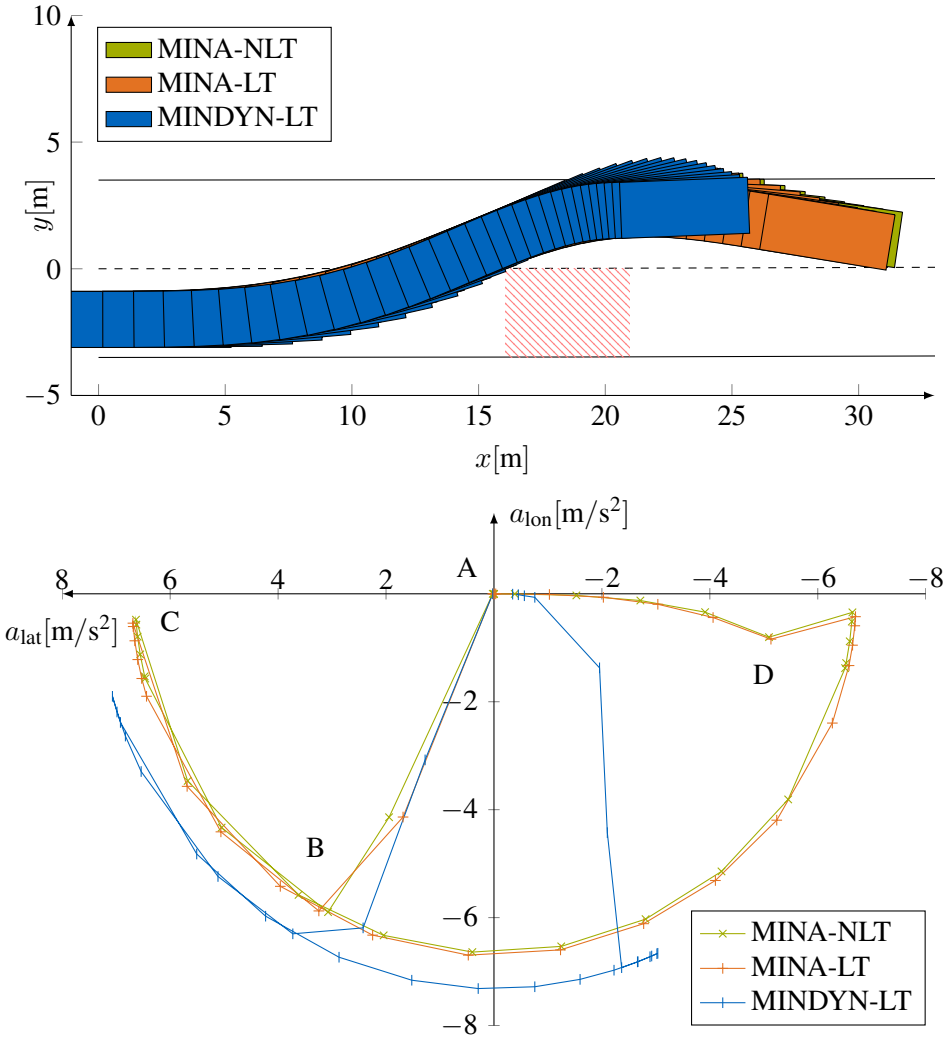


Figure 5.4: Evasion trajectories with an initial velocity of $v_h = 15\text{m/s}$ and static obstacle (hatched area) [69]. Left: top-down view, right: traces of longitudinal and lateral acceleration. OCP variants: max-norm of acceleration with nonlinear tire (green), linear tire (orange), maximum dynamics without input constraints (blue).

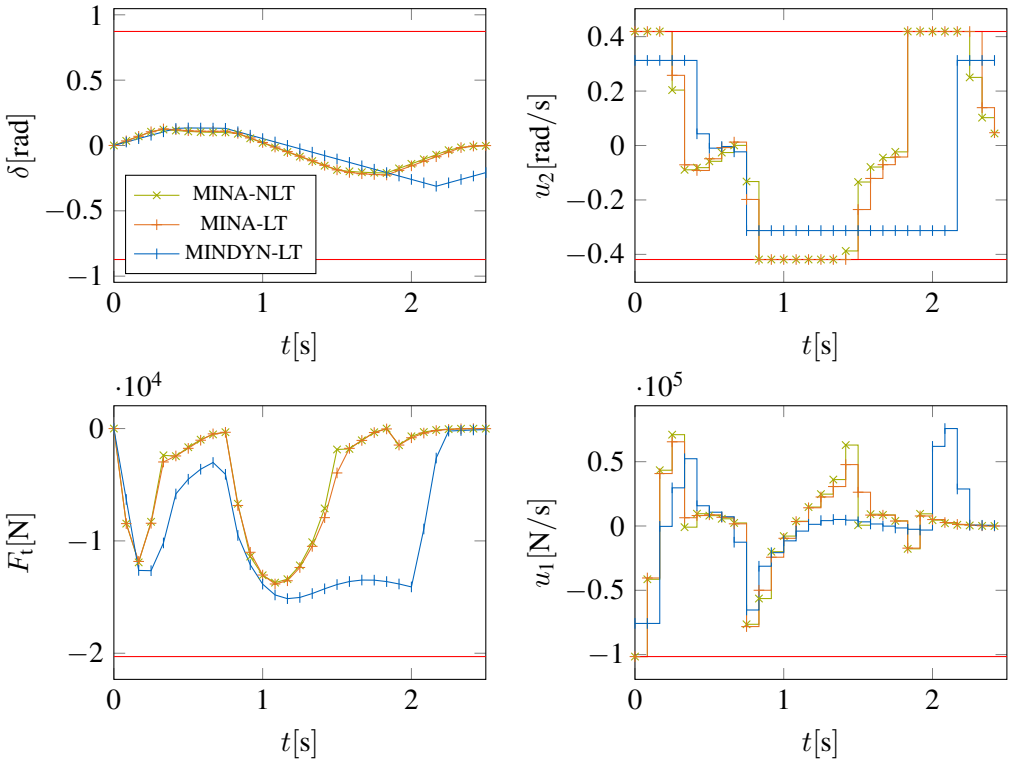


Figure 5.5: Evasion maneuver with $v_h = 15\text{m/s}$ and static obstacle: steering angle, tangential tire force, and control inputs

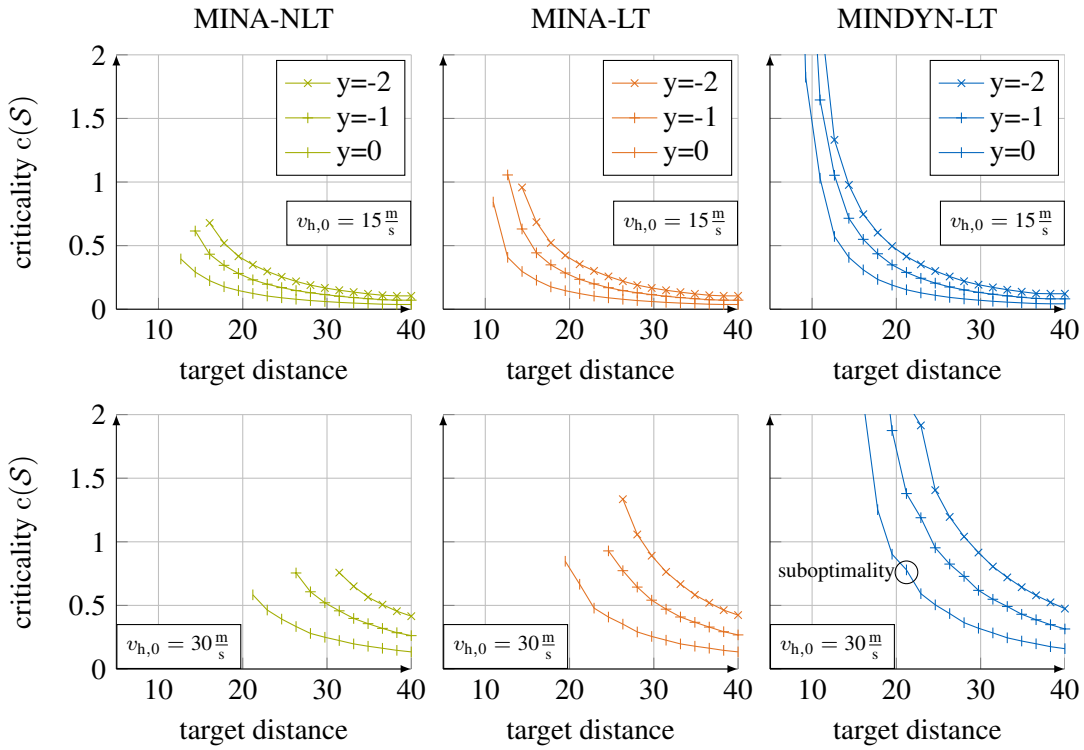


Figure 5.6: Criticality values on a straight road segment with a static obstacle and varying lateral offsets. From top to bottom: initial host vehicle velocities of 15m/s and 30m/s. From left to right: alternative OCP formulations as summarized in Table 4.1.

Chapter 6

Criticality Estimation via Supervised Machine Learning

In this section, we detail how to exploit the ability to generate criticality labels using optimal control by training a criticality estimator using supervised machine learning. In the context of active safety and autonomous driving, machine learning has led to breakthroughs in object detection for vehicles, bicycles, and pedestrians as well as for lane marking and traffic sign recognition. These applications have in common that they solve problems that are very hard to formulate using analytical, physics-oriented models and algorithms. Thus, one motivation for using a machine learning approach in a safety critical environment is the lack of a simpler analytical approach that offers competitive performance. In the context of this thesis, the use of statistical models offers the ability to estimate the criticality of avoidance trajectories with curvilinear road boundaries without incurring the computational cost of solving a full-blown optimal control problem in an online algorithm. We can therefore hope to combine the precision of the OCP approach with the real-time capability of supervised machine learning.

The statistical learning approach to online criticality estimation is depicted in Figure 6.1. Typical for supervised learning, we distinguish between the offline or training phase of the criticality estimator and the online operation. For the offline training, we generate a dataset $\mathcal{D}_{\text{tr}}$ consisting of a feature vector $\mathbf{z}_i$ for each driving scene $\mathcal{S}_i$ and its criticality label c_i. We restrict ourselves to the MINA-LT version of the labeling OCP (see, Table 4.1) as it provides a good compromise between interpretability and feasibility.

The feature vectors $\mathbf{z}_i$ are derived from the scene tuples $\mathcal{S}_i$ using a feature extrac-

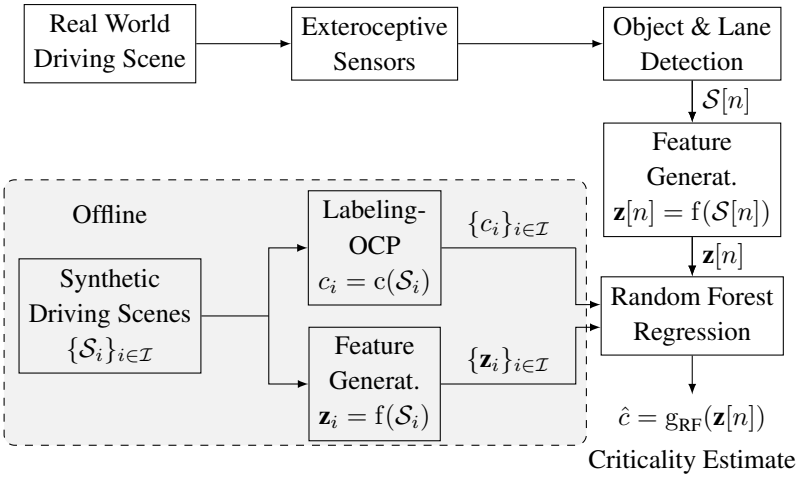


Figure 6.1: Overview of the statistical learning approach to criticality estimation using labels from optimal avoidance control

tion function $\mathrm{f} : \mathcal{X} \to \mathbb{R}^{S'}$, $\mathcal{S}_i \mapsto \mathbf{z}_i$. Here, $\mathcal{X}$ is the set of driving scenes and S' is the number of extracted features. We determine the type and number of features to extract using supervised feature extraction as detailed in Section 6.3.

After training the criticality estimator during the offline phase, the parameters of the estimator are kept fixed during the operation of the vehicle. During this online phase, the host vehicle observes its dynamic state and the state of the environment using a multitude of sensors, from accelerometers and gyroscopes for the dynamic state to cameras, radar, and lidar sensors. In currently available sensors and electronic control units, a highly developed signal process chain extracts interpretable information like object classes, relative velocities, or lane markings from the raw sensor data. In our application to criticality estimation, all elements necessary to construct the entries of the scene tuple $\mathcal{S}[n]$ have to be available on the BUS system of the vehicle. We detail the scene tuple extraction from BUS data in Appendix A. The feature extraction function f used during the offline training of the criticality estimator is also used online to compute the feature vector $\mathbf{z}[n]$. Based on this feature vector, the criticality is estimated by the criticality estimator.

A prerequisite for training the criticality estimators is a set of driving scenes $\{\mathcal{S}_i\}$ which we can label and exploit for feature extraction and learning. To this end, we define a probability density function which we use to generate synthetic driving scenes. This scene sampling method is described in Section 6.2. For the criticality estimator, we employ Random Forest Regression [56] which we motivate and detail in Section 6.1. In Section 7, the supervised criticality estimation approach is evaluated on synthetic data and on a resimulation real-world testing of dynamic collision avoidance on a test track.

6.1 Random Forest Regression

The Random Forest (RF) is a flexible and well-established machine learning technique that is equally appealing for classification, regression, and feature selection [56]. Random Forests are a so-called *ensemble* method, which derives its predictive accuracy from aggregating the predictions of large set Classification and Regression Trees (CART) [118], which are a form of decision tree (DT). Decision trees are invariant to strictly monotone scaling of the input features, resist the addition of uninformative features, are fast to construct and easily inspectable and interpretable [119, Ch. 10.7]. While losing the interpretability of DTs, Random Forests maintain all other favorable properties of DTs while offering significantly improved predictive accuracy compared to DTs. Another important consideration in favor of RF is the low number of tunable parameters that must be set for the training of the RF. Furthermore, an RF classification or regression model can be easily extended to include categorical features as inputs, e.g., such as traffic signs or the state of a traffic light for application in active safety. These properties make Random Forests an attractive choice for nonparametric (i.e., model-free) regression and classification tasks where meaningful features are available.

The Random Forest algorithm presented in the following was originally described in [56]. A derivation of the mathematical basis for RFs from first principles can be found in [67, Ch. 2.2]. We largely follow the derivation in that source.

6.1.1 Classification and Regression Trees

Classification and Regression Trees [118] divides the feature space $\mathcal{Z}$, which can be assumed to be $\mathbb{R}^N$ for simplicity, into a set of disjoint partitions $\mathcal{P}_n,\ \mathcal{P}_n \cap \mathcal{P}_{m\neq n} = \emptyset$ given by the partition index $n \in \mathcal{N}$. Thus, a CART assigns every element of the feature space to a unique partition n [67], which is expressed by the partition assignment function τ,

$$\tau : \mathcal{Z} \to \mathcal{N},\ \mathbf{z} \mapsto n. \tag{6.1}$$

To each element in the partition $\mathcal{P}_n$, a common output value $\nu(n)$ is assigned. The CART estimate for the label of a feature vector is therefore given by

$$\mathrm{g}_{\mathrm{DT}}(\mathbf{z}) = \nu(\tau(\mathbf{z})). \tag{6.2}$$

The partitioning of the feature space of a CART is represented by a binary tree as illustrated in Figure 6.2. To predict the label of a feature vector $\mathbf{z}$, we start at the root of the tree and, at each decision node, compare one dimension of the feature vector to the threshold value at that node. Depending on whether the entry of the feature vector is larger or smaller than the threshold, the corresponding branch is followed. This procedure is repeated until a leaf of the tree is reached which represents the partition

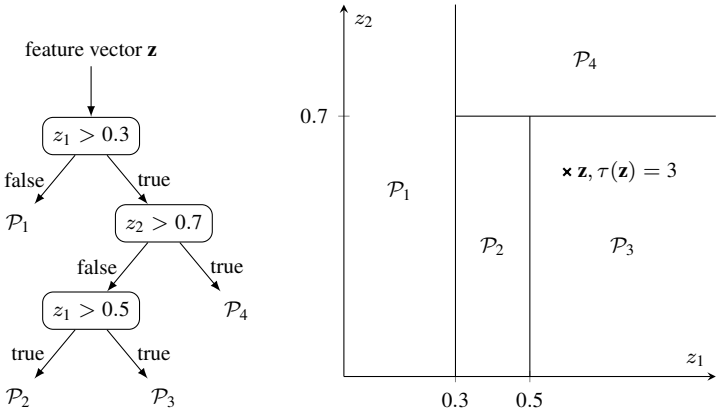


Figure 6.2: Left: binary tree representation of a CART example. Right: resulting partitioning of $\mathbb{R}^2$ and the assignment of a query feature vector $\mathbf{z}$ to partition $\mathcal{P}_3$.

n with the leaf value $\nu(n)$ which is the predicted label of the query vector $\mathbf{z}$. The learning task of a CART consists of inferring all aspects of the binary tree, namely,

- the feature used for each split,
- the threshold value of each split,
- whether to refine a partition by splitting it into smaller partitions,
- and the label estimate ν for each partition.

We denote by $\mathcal{L}_n$ the set of all labels (either discrete class labels or real-valued) of the training data that are assigned to a partition n by the CART, i.e.,

$$\mathcal{L}_n = \{y_i : \tau(\mathbf{z}_i) = n, (\mathbf{z}_i, y_i) \in \mathcal{D}_{\text{tr}}\}, \tag{6.3}$$

where $\mathcal{D}_{\text{tr}}$ is the training data set. For regression with a mean squared error loss, the label estimate of a partition n is the mean value of $\mathcal{L}_n$, i.e.,

$$\nu(n) = \frac{1}{|\mathcal{L}_n|} \sum_{c_i \in \mathcal{L}_n} c_i. \tag{6.4}$$

For classification, the estimate $\nu(n)$ is given by the majority class label in the partition, which writes as,

$$\nu(n) = \arg\max_j \left|\{i : \tilde{c}_i = j, \tilde{c}_i \in \mathcal{L}_n\}\right|. \tag{6.5}$$

The joint optimization of the feature space partitioning and partition estimates ν is computationally intractable [119]. Instead of performing such a joint optimization,

the CART algorithm employs a greedy splitting strategy. This means that the decision how to refine a partition into smaller partitions is only based on the training data inside the partition and that each splitting decision maximizes the immediate improvement in the prediction performance without considering splitting decisions further down the tree.

To refine a partition $\mathcal{P}_n$ into two sub partitions $\mathcal{P}_\mathrm{a}$ and $\mathcal{P}_\mathrm{b}$, the scalar element z_d of the feature vector that is used for the binary split, as well as the splitting threshold s has to be determined. The optimal splitting dimension d^* and threshold s^* minimize the splitting criterion [119],

$$d^*, s^* = \arg\min_{d,s} \left(\min_{\nu_\mathrm{a}} \sum_{c_i \in \mathcal{L}_\mathrm{a}(d,s)} (c_i - \nu_\mathrm{a})^2 + \min_{\nu_\mathrm{b}} \sum_{c_i \in \mathcal{L}_\mathrm{b}(d,s)} (c_i - \nu_\mathrm{b})^2 \right), \quad (6.6)$$

where the sub partitions $\mathcal{L}_\mathrm{b}$ and $\mathcal{L}_\mathrm{a}$ depend on the splitting dimension and threshold and the inner mean squared error optimizations are solved by (6.4). Applied to classification problems, the splits have to minimize a so-called node impurity measure $Q(\mathcal{P})$ of the sub partitions $\mathcal{P}_\mathrm{a}$ and $\mathcal{P}_\mathrm{a}$,

$$d^*, s^* = \arg\min_{d,s} \left(\min_{\nu_\mathrm{a}} Q(\mathcal{L}_\mathrm{a}(d,s) + \min_{\nu_\mathrm{b}} Q(\mathcal{L}_\mathrm{b}(d,s) \right). \quad (6.7)$$

The impurity measure quantifies the diversity of the class labels within the partition. Well known choices are the misclassification error, the Gini index and cross-entropy [119, Ch. 9.3].

The recursive splitting of the feature space partitions is stopped once each partition contains only a certain number $n_{\text{split,min}}$ of training samples, which is a tuning parameter. It is typical to choose a small number for $n_{\text{split,min}}$ and then apply pruning strategies (with their own set of parameters) to the grown CART to avoid overfitting. The reader is referred to [119, Ch. 9.2.2] for an overview of these pruning strategies.

6.1.2 The Bias-Variance Trade-off for Random Forest Regression

To estimate the generalization of an estimator, we consider the mean squared prediction error at query point $\mathbf{z}$. The query point $\mathbf{z}$ and its label c are not part of the training set $\mathcal{D}_\text{tr}$. The estimate at $\mathbf{z}$ is given by $\mathrm{g}(\mathbf{z}, \mathcal{D}_\text{tr})$, where $\mathrm{g}(\cdot, \mathcal{D}_\text{tr})$ is a regression estimator fitted to the training data set $\mathcal{D}_\text{tr}$. The random variables must be considered when estimating the generalization error are the random data set $\mathcal{D}_\text{tr}$ and the random label c. It is further assumed that the label depends on a deterministic but unknown function f and a white Gaussian noise term ϵ with variance σ_ϵ^2 and zero mean such that $c = f(\mathbf{z}) + \epsilon$. For convenience, we abbreviate the estimate at $\mathbf{z}$ as $\mathrm{g}(\mathbf{z}, \mathcal{D}_\text{tr}) = \hat{c}$.

The generalization is measured by the expected value of the squared prediction error with respect to the random variables $c, \mathcal{D}_{\text{tr}}$, i.e.,

$$\begin{aligned}\mathrm{E}_{c,\mathcal{D}_{\text{tr}}|\mathbf{z}}\left[(c-\hat{c})^2\right] &= \mathrm{E}\left[c^2\right] + \mathrm{E}\left[\hat{c}^2\right] - 2\,\mathrm{E}\left[c\hat{c}\right], \\ &= \mathrm{Var}\left[c\right] + \mathrm{E}\left[c\right]^2 + \mathrm{Var}\left[\hat{c}\right] + \mathrm{E}\left[\hat{c}\right]^2 - 2\,\mathrm{E}\left[c\hat{c}\right], \\ &= \sigma_\epsilon^2 + \mathrm{Var}\left[\hat{c}\right] + \mathrm{E}\left[c\right]^2 + \mathrm{E}\left[\hat{c}\right]^2 - 2\,\mathrm{E}\left[(f(\mathbf{z})+\epsilon)\hat{c}\right],\end{aligned}$$

where we have used that fact that for any random variable X we have $\mathrm{E}\left[X^2\right] = \mathrm{Var}\left[X\right] + \mathrm{E}\left[X\right]^2$. By noting that $\mathrm{E}\left[c\right] = f(\mathbf{z})$ and that the label noise ϵ is uncorrelated to the estimate $\hat{c}$, we can further rewrite this expression to

$$\begin{aligned}\mathrm{E}_{c,\mathcal{D}_{\text{tr}}|\mathbf{z}}\left[(c-\hat{c})^2\right] &= \sigma_\epsilon^2 + \mathrm{Var}\left[\hat{c}\right] + f(\mathbf{z})^2 + \mathrm{E}\left[\hat{c}\right]^2 - 2f(\mathbf{z})\,\mathrm{E}\left[\hat{c}\right], \\ &= \sigma_\epsilon^2 + \mathrm{Var}\left[\hat{c}\right] + (f(\mathbf{z}) - \mathrm{E}\left[\hat{c}\right])^2, \\ &= \sigma_\epsilon^2 + \mathrm{Var}\left[\hat{c}\right] + (\mathrm{E}\left[f(\mathbf{z}) - \hat{c}\right])^2, \\ &= \sigma_\epsilon^2 + \mathrm{Var}\left[\hat{c}\right] + \mathrm{Bias}[\hat{c}]^2. \end{aligned} \tag{6.8}$$

This is the famous bias-variance decomposition of the generalization error [119]. It consists of the variance of the estimator, its squared bias, and the label noise variance σ_ϵ^2, which poses a lower bound on the achievable generalization error.

An estimator with a low model complexity tends to have a large bias term as it is not able to adequately approximate the available data. Consider, for example, the constant estimator, $\hat{c} = \text{const}$, which has the lowest possible complexity. It will have a high bias term since it does not fit to the data and it has a variance term of zero. On the other hand, if an estimator has a high complexity, it can fit to the data to a high degree, reducing the bias term. It will also be more susceptible to the random sample realization of the training data $\mathcal{D}_{\text{tr}}$ and can fit to the label noise ϵ, which increases the variance term [119].

Analyzing the bias-variance tradeoff for an ensemble of CART learners gives an intuition about the appeal of ensemble methods in general and the Random Forest in particular. The regression function for an ensemble of N_{L} CARTs learners $\mathrm{g}_{\text{DT},l}$ is given by the mean of the ensemble predictions, i.e., $\hat{c} = \frac{1}{N_{\text{L}}}\sum_l \mathrm{g}_{\text{DT},l}(\mathbf{z})$. Inserting into the bias-variance expression (6.8) yields

$$\mathrm{E}_{c,\mathcal{D}_{\text{tr}}|\mathbf{z}}\left[(c-\hat{c})^2\right] = \sigma_\epsilon^2 + \mathrm{Var}\left[\frac{1}{N_{\text{L}}}\sum_l \mathrm{g}_{\text{DT},l}(\mathbf{z})\right] + \mathrm{Bias}\left[\frac{1}{N_{\text{L}}}\sum_l \mathrm{g}_{\text{DT},l}(\mathbf{z})\right]^2. \tag{6.9}$$

The bias term of the generalization error is reduced by leaving the CART base-learners unpruned, allowing them to tightly fit to the data which reduces their bias term. Compared to the variance of a single CART learner, the variance of the Random Forest

is reduced by averaging over the predictions of the ensemble. A prerequisite for the successful reduction of the variance term is that the base-learners in the ensemble are uncorrelated [56]. The Random Forest algorithm achieves a decorrelation of its base-learners by the combination of bootstrap aggregation and random feature selection, which are explained in the following.

6.1.3 Bootstrap Aggregation and Random Feature Selection

Random Forests [56] combine the concepts of bootstrap aggregation or *bagging* and random feature selection to improve upon the prediction performance and robustness of a single CART. To predict the label of a feature vector, it aggregates the predictions of the individual base learners using either the mean prediction of the base learners for regression or a majority vote for classification.

Each individual CART $\mathrm{g}_{\mathrm{DT},l}$ is trained on a so-called bootstrap sample. A bootstrap sample $\mathcal{D}_l$ is obtained from the training data set $\mathcal{D}_{\mathrm{tr}}$ by resampling $M = |\mathcal{D}_{\mathrm{tr}}|$ times with replacement. Training each CART on individual bootstrap samples improves the generalization of the random forest ensemble and allows to estimate the correlation of the base learners as well as the generalization performance of the RF ensemble [56, Ch. 3.1].

In addition to the random bootstrap sampling, random feature selection is used during the training of each CART. This means, that each splitting decision can only use a random subset of size $n_{\mathrm{features}} < S'_{\mathrm{all}}$ of all S'_{all} elements of the feature vector $\mathbf{z}$. Using a random subset of features makes the Random Forest robust to noise and uninformative features. Furthermore, it decorrelates - in tandem with bootstrap sampling - the individual CART learners. This decorrelation of the base learners reduces the variance of the prediction error as discussed above.

The random subset of features $\mathcal{I}_d$ that is considered for a splitting decision depends on the random variable θ_{split} which is represented by the urn model of drawing S' times without replacement from an urn that contains the feature indices. For a single split of a single CART in the RF ensemble, the randomized splitting criterion can now be written as

$$\begin{aligned} \mathcal{I}_d &= \{d_1, \ldots, d_{S'}\} \propto \theta_{\mathrm{split}} \\ d^*, s^* &= \underset{d \in \mathcal{I}_d, s}{\arg\min} \left(\min_{\nu_{\mathrm{a}}} \sum_{c_i \in \mathcal{L}_{\mathrm{a}}(d,s)} (c_i - \nu_{\mathrm{a}})^2 + \min_{\nu_{\mathrm{b}}} \sum_{c_i \in \mathcal{L}_{\mathrm{b}}(d,s)} (c_i - \nu_{\mathrm{b}})^2 \right), \end{aligned} \tag{6.10}$$

where, as with the default CART, $\mathcal{L}_{\mathrm{b}}$ and $\mathcal{L}_{\mathrm{a}}$ denote the sub partitions that depend on the splitting dimension and threshold and the inner mean squared error optimizations are solved by (6.4).

This splitting criterion is applied recursively for each CART in the ensemble until each tree is fully grown, i.e., until all partitions contain only a single training sample.

Furthermore, the decision trees not pruned after training. Each single CART in the ensemble is therefore susceptible to overfitting the data. However, the Random Forest still achieves a low prediction bias by aggregating the predictions of all base learners. In [56], this observation is confirmed by theoretical and empirical study.

Formally, the prediction of a random forest with ensemble size N_L writes as

$$\mathbb{R} \ni \mathrm{g}_\mathrm{RF}(\mathbf{z}) = \frac{1}{L} \sum_{l=1}^{N_\mathrm{L}} \mathrm{g}_{\mathrm{DT},l}(\mathbf{z}, \mathcal{D}_l, \theta_{\mathrm{split,l}}), \tag{6.11}$$

where $\mathrm{g}_{\mathrm{DT},l}(\mathbf{z}, \mathcal{D}_l, \theta_{\mathrm{split,l}})$ denotes the prediction of the l-th CART, $\mathcal{D}_l$ is the random bootstrap sample given to each CART for training and $\theta_{\mathrm{split,l}}$ is the random variable for input selection at each split.

In order to obtain a class prediction, the majority class among all N_L predictions is chosen. With N possible output classes $\tilde{c}_1, ..., \tilde{c}_N$, the RF prediction is

$$\{\tilde{c}_1, ..., \tilde{c}_N\} \ni \mathrm{g}_\mathrm{RF}(\mathbf{z}) = \underset{n \in 1,...,N}{\arg\max} \{V_1, \ldots, V_N\}, \tag{6.12}$$

where V_n indicates the number of learners $\mathrm{g}_{\mathrm{DT},l}(\mathbf{z}, \mathcal{D}_l, \theta_{\mathrm{split,l}})$ of the ensemble that predicted the n-th class.

Uncertainty measures and percentiles Ensemble predictions can also be used to obtain a measure of uncertainty of a prediction in terms of the prediction variance, i.e.,

$$\mathrm{Var}\, \hat{c}_i = \frac{1}{L-1} \sum_{l=1}^{N_\mathrm{L}} (\mathrm{g}_{\mathrm{DT},l}(\mathbf{z}_i, \mathcal{D}_l, \theta_{\mathrm{split,l}}) - \hat{c}_i))^2. \tag{6.13}$$

The uncertainty of a class prediction can be quantified by the relative frequency of the majority vote,

$$P = \frac{V_n}{N_\mathrm{L}}, \tag{6.14}$$

which gets closer to one with a higher prediction certainty of the RF. These measures can be used to reject the activation of an active safety system if the variance of the estimate is below a certain threshold leading to increased robustness of the system. We can also use the ensemble to estimate percentiles of the criticality, that is,

$$\hat{c}_p = \underset{\hat{c}}{\arg\min}\, \hat{c}, \text{ s.t. } |\{\hat{c}_l : \hat{c}_l \leq \hat{c}_j\}| \geq p N_\mathrm{L}, \tag{6.15}$$

where N_L is the ensemble size and $\hat{c}_l,\ l = 1, ..., N_\mathrm{L}$ are the predictions of the individual learners $\mathrm{g}_{\mathrm{DT},l}$. The ensemble predictions of an RF can also be used to approximate the full conditional distribution of the predicted variable as proposed in [120] as the Quantile Regression Forest.

6.1.4 Out-of-bag error estimates

Each bootstrap sample $\mathcal{D}_l$ contains about two thirds of the unique feature-label pairs of the original training data set $\mathcal{D}_{\text{tr}}$, since the bootstrap data is drawn from the training data with replacement. Therefore, it is possible to estimate the generalization error of the Random Forest by measuring the so-called out-of-bag prediction error of each individual learner. An OOB estimate for the label of a training sample that is part of the training set is obtained from the prediction of all CARTs of the RF ensemble whose bootstrap sample did not contain the training sample, i.e.,

$$\begin{aligned} \mathrm{g}_{\text{oob}}(\mathbf{z}_i; \mathcal{D}_{\text{tr}}) &= \frac{1}{N_{oob}(\mathbf{z}_i)} \sum_{l:\mathbf{z}_i \notin \mathcal{D}_l} \mathrm{g}_{\text{DT},l}(\mathbf{z}_i; \mathcal{D}_l, \theta_{\text{split,l}}), \\ N_{oob}(\mathbf{z}_i) &= |\{\mathcal{D}_l : \mathbf{z}_i \notin \mathcal{D}_l\}| \end{aligned} \tag{6.16}$$

where $N_{oob}(\mathbf{z}_i)$ is the number of trees that did not have $(\mathbf{z}_i, c_i)$ as an element of their bootstrap training sample $\mathcal{D}_l$. The authors of [121] have shown empirically that the out-of-bag estimate of the generalization is as accurate as the error estimate of predicting the labels of a test set the same size as the training set [56].

6.2 Stochastic Scene Sampling

For any form of machine learning, a set of training and test samples of feature vectors $\mathbf{z}_i$ is required for the training and evaluation of the statistical learner. In our case, the feature vectors are extracted from scene tuples $\mathcal{S}_i$ which contain a high-level description of the host vehicle state and the state of the environment, see Section 2.2 for details.

A typical assumption in statistical learning is that the training samples of feature vectors and labels $\{\mathbf{z}_i, c_i\}_{i=1,\ldots,M}$ are independent and identically distributed samples of random variables with a joint probability density function (PDF) $p(\mathbf{z}, c)$, where the PDF is typically unknown. Applied to criticality estimation in active safety, an ideal training set would consist of detailed recordings of real-world driving scenes with a mix of uncritical, stable driving, near-crash and crash events. The closest to this ideal are naturalistic driving studies where the cars of test subjects are equipped with data recording equipment for a prolonged period of time - up to more than two years in the case of the SHRP2 study [122]. Fortunately, near-crashes and crashes are rare events compared to most of the safe and crash-free everyday driving. This lack of detailed recordings of near-crash and crash events motivates the following derivation of a synthetic probability distribution for the scene tuples $\mathcal{S}_i$ from a-priori assumptions.

The goal of the stochastic scene model is to generate rear-end collision scenes with intuitively reasonable lane geometries and dynamic states that lead to criticality

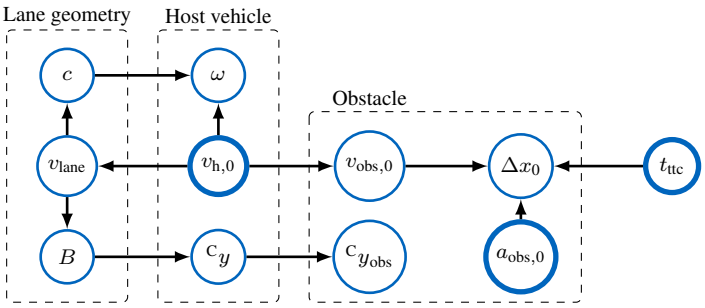


Figure 6.3: Stochastic variables and conditional dependence structure for the stochastic sampling of longitudinal collision scenes. Thick nodes: independent variables.

values ranging from uncritical to highly critical and unavoidable. We model the road geometry, the host vehicle state, the time-to-collision, and the obstacle state as stochastic variables. In order to generate a rear-end collision scene, we draw realizations or *samples* from these stochastic variables.

The conditional dependence structure of the stochastic scene sampling is illustrated by the graphical model in Figure 6.3. Here, an arrow going from a variable A to B indicates that the distribution of B depends on the realization of A. To generate random samples, we first sample from the independent variables (marked thick in Figure 6.3) $v_{\text{h},0}$, t_{ttc}, and $a_{\text{obs},0}$. Then, incrementally, we draw samples from those stochastic variables that depend on the sample realizations that have already been drawn. For example, we can only draw a sample of the curvature $c_i \sim \mathrm{p}_{c|v_{\text{lane}}}(c)$ if a speed limit has already been drawn.

For simplicity, we use only uniform probability distributions in the following. More elaborate distribution types like the truncated Gaussian distribution could be used, but they would require parameter fitting, e.g., by analyzing naturalistic driving data of driving simulator studies.

The independent variable time-to-collision is chosen as uniformly distributed, i.e.,

$$t_{\text{ttc},i} \sim \mathrm{p}_{t_{\text{ttc}}}(t_{\text{ttc}}) = \mathcal{U}(0.5\text{s}, 2\text{s}) \tag{6.17}$$

where $\mathcal{U}(a, b)$ is the uniform distribution on the interval $[a, b]$. Since the time-to-collision is a common and simple surrogate value for the criticality of a driving scene (see, e.g., [123]), the upper and lower bounds are chosen to lead to a large number of critical scenarios without resulting in a majority of unavoidable collisions.

The criticality estimation should work equally well for a range of host vehicle velocities. Therefore, $v_{\text{h},0}$ is also modeled as an independent, uniformly distributed variable, i.e.,

$$v_{\text{h},0,i} \sim \mathrm{p}_{v_{\text{h},0}} = \mathcal{U}(5, 35)\text{m/s}. \tag{6.18}$$

Table 6.1: Conditional probability of the design speed v_{lane} of a lane segment

$v_{\text{h},0}[\text{km/h}]$	$0 < v_{\text{h},0} \leq 60$	$60 < v_{\text{h},0} \leq 100$	$100 < v_{\text{h},0}$
$P(v_{\text{lane}} = 50 \mid v_{\text{h},0})$	$\frac{1}{3}$	0	0
$P(v_{\text{lane}} = 80 \mid v_{\text{h},0})$	$\frac{1}{3}$	$\frac{1}{2}$	0
$P(v_{\text{lane}} = 120 \mid v_{\text{h},0})$	$\frac{1}{3}$	$\frac{1}{2}$	1

For the design of the probability distribution of the road geometry, we take inspiration from road construction laws in the German road construction regulation [76]. There, the *Entwurfsgeschwindigkeit* or designated speed v_{lane} of the road segment determines the allowable curvatures, and rate of change of the curvatures of the segment.

Note that for a given host vehicle velocity $v_{\text{h},0}$, only certain ranges of design speeds are likely, assuming that the host vehicle does not grossly exceed the design speed of the road segment. We distinguish between slow, medium, and high design speeds and host vehicle speeds. Given a slow vehicle speed, the design speed may vary from slow to high, since it is plausible to drive slowly on a segment with a high design speed, e.g., due to congestion. Similarly, for a moderate vehicle speed, the design speed may be moderate or high and for a high host vehicle speed, the design speed may only be high. This reasoning motivates the conditional probabilities of the design speed listed in Table 6.1.

For road segments with higher design speeds, the maximum curvature decreases in order to limit the lateral acceleration during lane keeping. Thus, given a design speed, we define the maximum curvature of the lane segment as

$$c_{\text{max}} = \frac{1}{r_{\text{min}}}, \quad r_{\text{min}} = \begin{cases} 80\text{m} & \text{if } v_{\text{lane}} = 50\text{km/h} \\ 250\text{m} & \text{if } v_{\text{lane}} = 80\text{km/h} \\ 720\text{m} & \text{if } v_{\text{lane}} = 120\text{km/h} \end{cases}. \tag{6.19}$$

These values are taken from German regulation for road construction [76][1]. The curvature is modeled as uniformly distributed between zero and its upper bound,

$$c_i \sim \mathrm{p}_{c|v_{\text{lane}}}(c) = \mathcal{U}(0, c_{\text{max}}). \tag{6.20}$$

Likewise, the lane widths $B_l = B_r = B$ are uniformly distributed, i.e.,

$$B_i \sim \mathrm{p}_{B|v_{\text{lane}}}(B) = \mathcal{U}\left(B_{\text{min}}, B_{\text{max}}\right), \tag{6.21}$$

[1] Only the numerical values are taken from [76]; the proposed sampling scheme is independent of the regulation.

where the upper and lower bounds again depend on the design speed of the lane segment, i.e.,

$$(B_{\text{min}}, B_{\text{max}}) = \begin{cases} (2.75\text{m}, 3.25\text{m}) & \text{if } v_{\text{lane}} = \ 50\text{km/h} \\ (3.5\text{m}, 4.0\text{m}) & \text{if } v_{\text{lane}} \in \{80\text{km/h}, 120\text{km/h}\} \end{cases}. \tag{6.22}$$

In [76, Table 7], bounds on the clothoid rate of change of the curvature of the segment are given by

$$\kappa \in \left[\frac{1}{A_{\text{max}}^2}, \frac{1}{A_{\text{min}}^2}\right], \quad A_{\text{min}} = \frac{1}{3c}, \ A_{\text{max}} = \frac{1}{c}. \tag{6.23}$$

In our simulations, we set the rate of change of the curvature to the center of the allowed interval, i.e.,

$$\kappa = 0.5\frac{1}{A_{\text{max}}^2} + 0.5\frac{1}{A_{\text{min}}^2}. \tag{6.24}$$

Since the host vehicle has a width W, we limit the lateral position in the uniform distribution

$${}^{\text{C}}y_i \sim \text{p}_y\left({}^{\text{C}}y\right) = \mathcal{U}(y_{\text{min}}, y_{\text{max}}). \tag{6.25}$$

Here, the upper and lower bounds of the lateral offset $y_{\text{min}}, y_{\text{max}}$ are $B_i - 0.5W$ and $-B_i + 0.5W$, respectively. Assuming that the host vehicle follows the lane curvature, the initial yaw rate of the host vehicle is a function of the initial curvature c_i, lateral offset ${}^{\text{C}}y_i$, and velocity $v_{\text{h},0,i}$,

$$\omega_i = \frac{v_{\text{h},0,i}}{\rho}, \quad \rho = \frac{1}{c_i} - {}^{\text{C}}y_i, \tag{6.26}$$

where ρ is the initial turn radius of the host vehicle.

It remains to determine the position, velocity, and acceleration of the obstacle. Since we want to generate driving scenes that lead to a collision, it is reasonable to upper-bound the obstacle velocity by the host vehicle velocity. The conditional distribution of the obstacle speed is therefore chosen as

$$v_{\text{obs},i} \sim \text{p}_{v_{\text{obs},0}|v_{\text{h},0}}(v_{\text{obs},0}) = \mathcal{U}(0, v_{\text{h},0,i}). \tag{6.27}$$

The obstacle is placed in the same lane as the host vehicle and blocks the whole lane, see the restricted area in Figure 2.2. Finally, the obstacle acceleration is chosen as an independent variable with the distribution

$$a_{\text{obs},i} \sim \text{p}_{a_{\text{obs}}}(a_{\text{obs}}) = \mathcal{U}(-6\text{ms}^{-2}, 0), \tag{6.28}$$

where the deceleration limit of -6ms^{-2} represents are hard braking maneuver.

Given realizations of the time-to-collision as well as the obstacle velocity and acceleration, we compute the initial distance of the obstacle as the distance that leads to the sampled time-to-collision value, i.e.,

$$t_{\text{ttc},i} = t_{\text{ttc}}(\Delta x_i, v_{\text{h},0,i}, v_{\text{obs},i}, a_{\text{obs},i}) \tag{6.29}$$

The time-to-collision function $t_{\text{ttc}}(\cdot)$ for the solution of this implicit equations is given in (6.41).

6.2.1 Training and Validation Datasets

We use the stochastic model of the scene probabilities to draw random driving scenes which we can use for feature selection, training, and validation of the Random Forest criticality estimator. First, we draw M scene tuples $\mathcal{S}_i, i = 1, \ldots, M$ by sampling the probability distribution functions of the scene geometry and kinematics given in (6.17) to (6.29). Then, the criticality of each scene is added as a label by solving the Optimal Control Problem (OCP) (4.47) as described in Section 4. The labeled dataset is denoted as

$$\mathcal{D} = \{(\mathcal{S}_i, \mathrm{c}(\mathcal{S}_i)\}_{i=1,\ldots,M}, , \tag{6.30}$$

where $\mathrm{c}(\mathcal{S}_i)$ is the optimal value of the MINA-LT version of the OCP (4.47).

For training and feature selection described in Section 6.4, we generated 7000 driving scenes. The labeling OCP converged to an optimal value for $M_{\text{tr}} = 6681$ of those scenes. Similarly, in order to measure the performance of the trained estimators, a validation dataset $\mathcal{D}_{\text{val}}$ of size $M_{\text{val}} = 1906$ (from 2000 scene tuples) is used that is excluded from feature selection and training, i.e.,

$$\mathcal{D}_{\text{val}} \cap \mathcal{D}_{\text{tr}} = \emptyset. \tag{6.31}$$

In Figure 6.4, we plot empirical cumulative density functions (ECDF) of the lane geometry and kinematics of the of the training and validation samples. Given a set of N samples a_i of a random Variable A, the ECDF is given by,

$$\mathrm{F}(a) = \frac{1}{N}|\{a_i : a_i \leq a\}|. \tag{6.32}$$

The ECDFs confirm that the geometric and kinematic properties of the driving scenes are evenly distribution on the whole domain of considered driving states without a concentration of samples on any particular area of state space. The only variable that shows a significant concentration is the distance Δx between host and obstacle. This is because the Δx is chosen as a function of the relative velocity and acceleration in order to achieve a desired time-to-collision t_{ttc}.

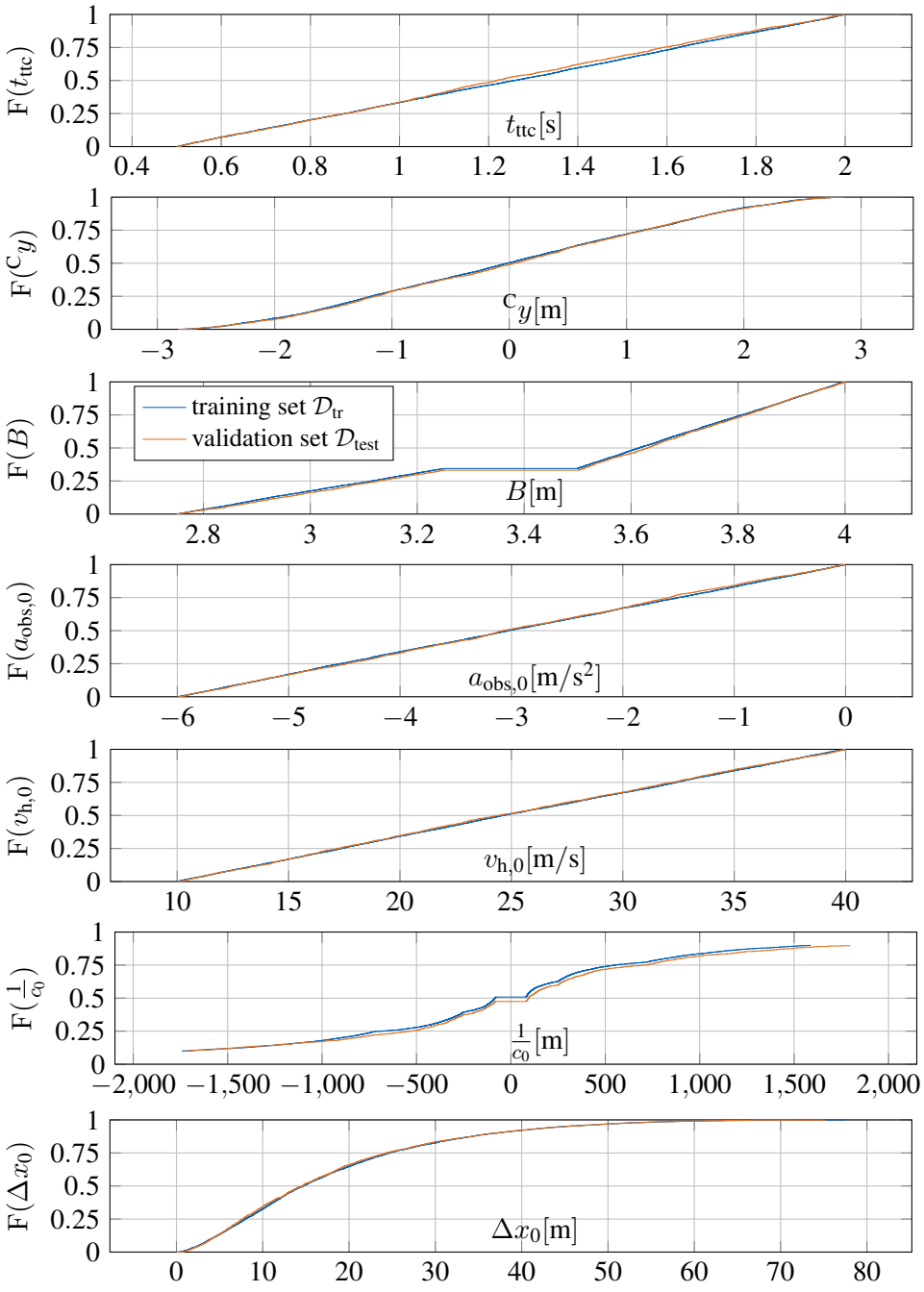


Figure 6.4: Empirical cumulative distribution functions of the training and validation data generated by sampling from the a-priori distribution for collision scenes, ref. Chapter 6.2

Table 6.2: Features for criticality prediction

Feature Type	Meaning	Variable, Formula
Scene data	Lateral Offset	$z_0 = \Delta y$
	EGO velocity	$z_1 = v_{\mathrm{h},0}$
	Lane Curvature	$z_2 = c$
	Lane Width, left	$z_3 = B_{\mathrm{l}}$
	Lane Width, right	$z_4 = B_{\mathrm{r}}$
	Distance to OBS	$z_5 = \Delta x_0$
	OBS velocity	$z_6 = v_{\mathrm{obs},0}$
	OBS acceleration	$z_7 = a_{\mathrm{obs},0}$
Threat surrogates	time-to-collision	$z_8 = t_{\mathrm{ttc}}$, ref. (6.41)
	Lon. avoidance acceleration	$z_9 = a_{\mathrm{x}}$, ref. (6.54)
	Lat. avoidance acceleration	$z_{10} = a_{\mathrm{y}}$, ref. (6.57)
Timings	Distance over speed difference	$z_{11} = \frac{\Delta x_0}{\Delta v_0}$
Ratios	Lateral and longitudinal distance	$z_{12} = \frac{\Delta y}{\Delta x}$
	Offset over host velocity	$z_{13} = \frac{\Delta y}{v_{\mathrm{h},0}}$
Acceleration	initial radial acceleration	$z_{14} = \frac{v_{\mathrm{h},0}^2}{c_0}$

6.3 Feature Extraction

The performance of regression estimation is known to be highly dependent on the availability of informative features. For criticality prediction, any quantity that contains information about the traffic scene can be a feature, e.g., all elements of the scene tuple $\mathcal{S}$ and any (non-) linear transformation of those quantities. We generate three types of features. First are entries of the scene tuple $\mathcal{S}$, ref (2.12). Second, we compute so-called threat surrogates, i.e., simple predictors for the threat of a traffic scene. These are the time-to-collision (TTC) t_{ttc}, as well as the longitudinal and lateral avoidance accelerations a_{x} and a_{y}. Additionally, we compute the lateral acceleration necessary for lane keeping. The third type of features are simple combinations of entries of the scene tuples, e.g., distance ratios and timings. The generated features are summarized in Table 6.2. While most of the features in Table 6.2 are very simple, the threat surrogates time-to-collision and longitudinal and lateral avoidance acceleration warrant a derivation. Let the distance, speed-difference, and acceleration difference of

the obstacle and host vehicle at $t = 0$ be denoted as

$$\Delta x_0 = {}^{\mathrm{C}}x_{\mathrm{obs},0} - {}^{\mathrm{C}}x_0 - 0.5L, \tag{6.33}$$

$$\Delta v_0 = v_{\mathrm{obs},0} - v_{\mathrm{h},0}, \tag{6.34}$$

$$\Delta a_0 = a_{\mathrm{obs},0} - a_{\mathrm{lon},0}, \tag{6.35}$$

where L is the host vehicle length.

Time-to-collision

We compute the time-to-collision in one-dimension for constant accelerations. For the obstacle, we assume that it accelerates with a constant acceleration but only until it comes to a standstill. Thus, we distinguish between two cases. First, the collision may occur before the obstacle comes to a halt. The distance over time is given by the kinematic equation

$$\Delta x_1(t) = \Delta x_0 + \Delta v_0 t + 0.5\Delta a_0 t^2. \tag{6.36}$$

The collision occurs when the distance between the host and obstacle is zero. The TTC is the smallest positive root of the quadratic equation (6.36), i.e.,

$$t_1 = \min\{\tau_1, \tau_2 : \tau_i \geq 0\},\ \tau_{1,2} = \frac{-\Delta v_0 \pm \sqrt{\Delta v_0^2 - 2\Delta a_0 \Delta x_0}}{\Delta a_0}. \tag{6.37}$$

In the second case, the collision occurs after the obstacle has come to a halt. Here, we need to compute the time-to-standstill, i.e.,

$$\bar{t} = -\frac{v_{\mathrm{obs},0}}{a_{\mathrm{obs},0}}. \tag{6.38}$$

If $\bar{t} < t_1$, the time-to-collision is given by the time it takes the EGO vehicle to reach the position of the obstacle at the time-of-standstill with a constant EGO acceleration and a standing obstacle. The kinematic equation for $t \geq \bar{t}$ becomes

$$\Delta x_2(t) = \Delta x(\bar{t}) + \Delta v(\bar{t})(t - \bar{t}) - 0.5 a_{\mathrm{lon},0}(t - \bar{t})^2, . \tag{6.39}$$

As for the first case (6.37), the time-to-collision is obtained from the distance over time, i.e.,

$$t_2 = \min\{\tau_1, \tau_2 : \tau_i \geq \bar{t}, \Delta x_2(\tau_i) = 0\}, \tag{6.40}$$

The time-to-collision is therefore given as,

$$t_{\mathrm{ttc}} = \begin{cases} t_1, & \text{if } t_1 < \bar{t}, \\ t_2, & \text{otherwise.} \end{cases} \tag{6.41}$$

Note that the time-to-collision is only well-defined if the kinematics actually lead to a collision in finite time. This is ensured during the generation of driving scenes for simulation and training as described in Chapter 6.2. During online operation, a finite, positive TTC has to be used as a precondition for criticality estimation.

Longitudinal Avoidance Acceleration

The longitudinal avoidance acceleration is defined as the acceleration a_x with minimal absolute value that leads to a vanishing velocity difference between the obstacle and host vehicle at the time of contact. In other words, it is the acceleration of the host vehicle with least magnitude that avoids damage to the traffic participants. The computation of the avoidance acceleration is more involved than the that of the time-to-collision since the time of contact τ depends on the longitudinal acceleration. We assume that the host vehicle acceleration immediately jumps to the avoidance acceleration at the start of the prediction horizon. The acceleration difference between the host vehicle and the obstacle becomes,

$$\Delta a_0 = a_{\mathrm{obs},0} - a_\mathrm{x}, \tag{6.42}$$

the following conditions hold at the time of contact τ:

$$\Delta x(t = \tau) = 0, \tag{6.43}$$

$$\Delta v(t = \tau) = 0. \tag{6.44}$$

As for the computation of the time-to-collision, we enumerate the cases $\tau \leq \bar{t}$ and $\tau > \bar{t}$, with $\bar{t}$ defined in (6.38). Beginning with $\tau \leq \bar{t}$, we have two equations for the two unknowns τ_1 and $a_{\mathrm{x},1}$,

$$0 = \Delta x_0 + \tau_1 \Delta v_0 + 0.5\tau_1^2(a_{\mathrm{obs},0} - a_{\mathrm{x},1}) \tag{6.45}$$

$$0 = \Delta v_0 + \tau_1(a_{\mathrm{obs},0} - a_{\mathrm{x},1}). \tag{6.46}$$

Solving (6.46) for $(a_{\mathrm{obs},0} - a_{\mathrm{x},1})$, we obtain the expression

$$(a_{\mathrm{obs},0} - a_{\mathrm{x},1}) = -\frac{\Delta v_0}{\tau_1}, \tag{6.47}$$

and inserting into (6.45) yields the time of contact as

$$0 = \Delta x_0 + 0.5\tau_1 \Delta v_0, \tag{6.48}$$

$$\Rightarrow \tau_1 = -\frac{2\Delta x_0}{\Delta v_0}. \tag{6.49}$$

The corresponding avoidance acceleration is obtained by inserting τ_1 into (6.47):

$$a_{\mathrm{x},1} = a_{\mathrm{obs},0} + \frac{\Delta v_0}{\tau_1} = a_{\mathrm{obs},0} - 0.5\frac{\Delta v_0^2}{\Delta x_0},$$

In the case $\tau > \bar{t}$, we assume that the contact only occurs after the obstacle has come to a halt. The obstacle position is therefore fixed, and the host vehicle velocity also must become zero, i.e.,

$$x_\mathrm{obs}(\tau_2) = x_\mathrm{obs}(\bar{t}), \quad v_\mathrm{obs}(\tau_2) = v_{\mathrm{h},0} + a_{\mathrm{x},2}\tau_2 = 0. \tag{6.50}$$

The avoidance acceleration therefore computes as

$$a_{\mathrm{x},2} = -\frac{v_{\mathrm{h},0}}{\tau_2}, \tag{6.51}$$

where $v_{\mathrm{h},0}$ is the initial host vehicle velocity. The position of the host vehicle at the time of contact τ_2 is given by

$${}^{\mathrm{C}}x_{\mathrm{h}}(\tau_2) = {}^{\mathrm{C}}x_0 + 0.5L + \tau_2 v_{\mathrm{h},0} + 0.5\tau_2^2 a_{\mathrm{x},2} = {}^{\mathrm{C}}x_0 + 0.5L + 0.5\tau_2 v_{\mathrm{h},0}. \tag{6.52}$$

Since the position of the host vehicle front and the obstacle are equal at τ_2, we obtain the equation

$$x_{\mathrm{obs}}(\bar{t}) = {}^{\mathrm{C}}x_{\mathrm{h}}(\tau_2) = {}^{\mathrm{C}}x_0 + 0.5L + 0.5\tau_2 v_{\mathrm{h},0},$$

so the solution for the contact time and avoidance acceleration is

$$\tau_2 = \frac{x_{\mathrm{obs}}(\bar{t}) - {}^{\mathrm{C}}x_0 - 0.5L}{0.5 v_{\mathrm{h},0}}, \quad a_{\mathrm{x},2} = \frac{0.5 v_{\mathrm{h},0}^2}{{}^{\mathrm{C}}x_0 + 0.5L - x_{\mathrm{obs}}(\bar{t})}. \tag{6.53}$$

The longitudinal avoidance acceleration can now be expressed as

$$a_{\mathrm{x}} = \begin{cases} a_{\mathrm{obs},0} - 0.5\frac{\Delta v_0^2}{\Delta x_0} & \text{if } \tau_1 \leq \bar{t}, \\ \frac{0.5 v_{\mathrm{h},0}^2}{{}^{\mathrm{C}}x_0 + 0.5L - x_{\mathrm{obs}}(\bar{t})} & \text{otherwise.} \end{cases} \tag{6.54}$$

Lateral Avoidance Acceleration

The lateral avoidance acceleration a_{y} is computed under the assumption of constant longitudinal acceleration and no coupling between longitudinal and lateral dynamics. The avoidance problem is only treated in one dimension, which neglects the lateral acceleration due to the lane curvature. It is the lateral acceleration with minimal magnitude that shifts EGO sideways such that it can pass the obstacle. The assumption of constant acceleration and decoupled dynamics yield

$${}^{\mathrm{C}}y(t) = {}^{\mathrm{C}}y_0 + 0.5t^2 a_{\mathrm{y}}. \tag{6.55}$$

The time-to-collision t_{ttc} is the time when the distance between the front of the host vehicle and the rear end of the obstacle is zero, assuming constant longitudinal accelerations, ref (6.41). At that time, the lateral EGO coordinate ${}^{\mathrm{C}}y$ has to be equal to the coordinate of the obstacles left rear corner plus half of the EGO width W_{h} to avoid a collision, i.e.,

$${}^{\mathrm{C}}y(t_{\mathrm{ttc}}) = y_{\mathrm{obs},0} + 0.5W_{\mathrm{h}} = {}^{\mathrm{C}}y_0 + 0.5t_{\mathrm{ttc}}^2 a_{\mathrm{y}}. \tag{6.56}$$

This also assumes that the obstacle keeps its lane with a constant lateral offset $y_{\mathrm{obs},0}$ as described in Section 2.2. We can now solve (6.56) for the lateral avoidance acceleration, i.e.,

$$a_{\mathrm{y}} = \frac{y_{\mathrm{obs},0} - {}^{\mathrm{C}}y_0 + 0.5W_{\mathrm{h}}}{0.5t_{\mathrm{ttc}}^2}. \tag{6.57}$$

6.4 Feature and Model Selection

In the following, we detail the feature selection and parameter search procedures used for training the Random Forest criticality estimator. The goal of feature selection is to determine the subset of all available features that maximizes the generalization of the trained model, i.e., how many features are necessary and which features should be used. This can also be necessary due to the curse of dimensionality [124] which, for an increasing number of features, leads to a degradation of the approximation performance of an estimator on theoretical grounds, as well as an intractable size of the optimization problems that must be solved to parameterize an estimator. Apart from improving generalization, using as few features as possible can significantly improve the training and prediction times of the trained estimators.

Model selection is the process of determining the parameters of the ML algorithm which are kept fixed during the training phase. These parameters are often referred to as *hyper parameters*. They determine the complexity or flexibility of the functions that can be represented using the machine learning model. As a rule of thumb, if the approximating function is more complex than necessary for the dataset, the model tends to overfit to the training data and perform worse on unseen data, a phenomenon commonly referred to as *overfitting*. It is therefore the role of model selection, to determine the set of hyper parameters that balances the flexibility of the learner with the ability to generalize to data that is not part of the training set. A thorough statistical treatment of overfitting and generalization in the context of machine learning is provided, e.g., by statistical learning theory [125].

Feature selection and parameter search both require a score function that predicts the generalization error of the model for the given features and parameters. We use the so-called coefficient of determination (COD) [126]

$$
\begin{aligned}
r_2 &= 1 - \frac{\sum_{i=1}^{M} (c_i - \mathrm{g}(\mathbf{z}_i; \mathcal{D}_{\text{tr}}))^2}{\sigma_c^2}, \quad (\mathbf{z}_i, c_i) \in \mathcal{D}_{\text{test}}, \\
\sigma_c^2 &= \sum_{i=1}^{M} (c_i - \bar{c})^2, \quad \bar{c} = \frac{1}{M} \sum_{i=1}^{M} c_i,
\end{aligned}
\tag{6.58}
$$

where g is the regression function fitted to the training data $\mathcal{D}_{\text{tr}}$. The COD relates the amount of label variance that can be explained by the predictor variables of the feature vector to the variance of all labels. It is a popular goodness-of-fit measure that get closer towards the value one the better the regression function $\mathrm{g}(\mathbf{z})$ fits the data. As discussed in [126], there are multiple definitions of the COD in the literature, of which the above definition is recommended by the author of [126].

In order to estimate the generalization error of the Random Forest instead of the training error, the COD is evaluated for the out-of-bag (OOB) samples of each tree of

Table 6.3: Random Forest hyper parameters

(a) Hyperparameters used for greedy feature selection

Hyper parameter	Variable	Value
Ensemble size	n_{trees}	1000
Features per split	n_{features}	n_{sqrt}, see (6.64)
Min. nb. of samples per split	$n_{\text{split,min}}$	2
Max. tree depth	d_{max}	∞
Split criterion		MSE

(b) Hyperparameters used for grid search

Hyper parameter	Variable	Grid Values
Ensemble size	n_{trees}	$\{500, 1000, 1500\}$
Features per split	n_{features}	$\{n_{\text{sqrt}} - 1, n_{\text{sqrt}}, n_{\text{sqrt}} + 1\}$
Min. nb. of samples per split	$n_{\text{split,min}}$	2
Max. tree depth	d_{max}	∞
Split criterion		MSE

the RF ensemble, see (6.16). The out-of-bag estimate of the COD is given by

$$r_{2,\text{RF}} = 1 - \frac{\sum_{i=1}^{M} (c_i - \mathrm{g}_{\text{oob}}(\mathbf{z}_i; \mathcal{D}_{\text{tr}}))^2}{\sigma_c^2}, \tag{6.59}$$

where the Random Forest out-of-bag estimate $\mathrm{g}_{\text{oob}}(\mathbf{z}_i; \mathcal{D}_{\text{tr}})$ is defined in (6.16) and the label variance σ_c^2 is given in (6.58).

Evaluating the score of every possible combination of feature set and hyper parameters is prohibitive due to the exponential number of possible combinations and the associated computational effort. Therefore, it is necessary to use suboptimal heuristics for the parameter and feature selection. We proceed in two steps. First, we perform a greedy search for the optimal number and type of features extracted from the scene tuple. In the second step, we perform grid search of the hyper parameters for each of the unique feature sets determined with greedy forward selection. The hyper parameters for feature selection and grid search are listed in Tables 6.3a and 6.3b.

6.4.1 Greedy Forward Feature Selection

Feature selection determines a matrix $\mathbf{S}$ that selects a feature vector $\mathbf{z}(\mathcal{S}) \in \mathbb{R}^{S'}$ which contains a subset of the feature vector of all available features $\mathbf{z}_{\text{all}}(\mathcal{S})$ (given in Table

6.2), i.e.,

$$\mathbf{z}(\mathcal{S}) = \mathbf{S}\mathbf{z}_{\text{all}}(\mathcal{S}). \tag{6.60}$$

A selection matrix $\mathbf{S}$ is a wide matrix where each row is a unit vector and all rows are orthogonal. The set of selection matrices is given by

$$\mathcal{S} = \left\{ S' \leq S'_{\text{all}},\ \mathbf{S} \in \{0,1\}^{S' \times S'_{\text{all}}} : \mathbf{1}_{S'}^{\mathrm{T}} \mathbf{S} = \mathbf{1}_{S'_{\text{all}}}^{\mathrm{T}} \right\}, \tag{6.61}$$

where $\mathbf{1}_n$ is the all-ones vector of size n. The selection matrix is chosen such that it optimizes the generalization performance of the machine learning algorithm on a training dataset $\mathcal{D}_{\text{tr}}$ [57], i.e.,

$$\mathbf{S} = \underset{\mathbf{S} \in \mathcal{S}}{\arg\max}\, r_{2,\text{RF}}(\mathcal{D}_{\text{tr}}), \text{s.t. } \mathcal{D}_{\text{tr}} = \{\mathbf{S}\mathbf{z}_{\text{all}}(\mathcal{S}_i), c_i\} \tag{6.62}$$

where $r_{2,\text{RF}}$ is the score function given in (6.59) that measures the generalization of the regression function and $\mathcal{S}_i$ and c_i are the scene tuples and criticality labels, see (6.30). Greedy forward feature selection [127] is a feature selection heuristic that iteratively adds a feature to the feature vector of the last iteration such that the new feature vector maximizes the score function. This can be written as the iteration rule

$$\begin{aligned}
\mathbf{e}^{(m)} &= \underset{\mathbf{e} \in \{0,1\}^{S'_{\text{all}}}}{\arg\max}\, r_{2,\text{RF}}(\mathcal{D}_{\text{tr}}),\ \text{s.t.,} \\
\mathcal{D}_{\text{tr}} &= \{\mathbf{S}^{(m)}\mathbf{z}_{\text{all}}(\mathcal{S}_i), c_i\} \\
\mathbf{S}^{(1)} &= \mathbf{e}^{(1),\mathrm{T}},\ \mathbf{S}^{(m)} = \begin{bmatrix} \mathbf{S}^{(m-1)} \\ \mathbf{e}^{\mathrm{T}} \end{bmatrix} \in \mathcal{S}, \forall m \in 2, \ldots, S' < S'_{\text{all}}.
\end{aligned} \tag{6.63}$$

This procedure yields a sequence of selection matrices $\mathbf{S}^{(m)},\ m = 1, \ldots, S'$ that contains the indices of the most informative features.

Since Random Forests are noisy by design, the results of the greedy forward feature selection might vary from run to run. This element of noisiness is an inherent property of statistical learning and therefore not unique to Random Forests. It is also well known for cross-validation, see, e.g., [119], and should be taken into account during model search.

During greedy selection, we use a fixed set of RF hyper parameters as detailed in Table 6.3a. For the number of features per split n_{features}, we apply the formula

$$n_{\text{sqrt}} = \left\lceil \sqrt{\dim(\mathbf{z})} \right\rceil, \tag{6.64}$$

where $\dim(\mathbf{z})$ is the dimensionality of the feature vector and $\lceil \cdot \rceil$ is the rounding-up operation. Different versions of the square-root or the logarithm of the number of features are commonly recommended as a starting point for the number of features per

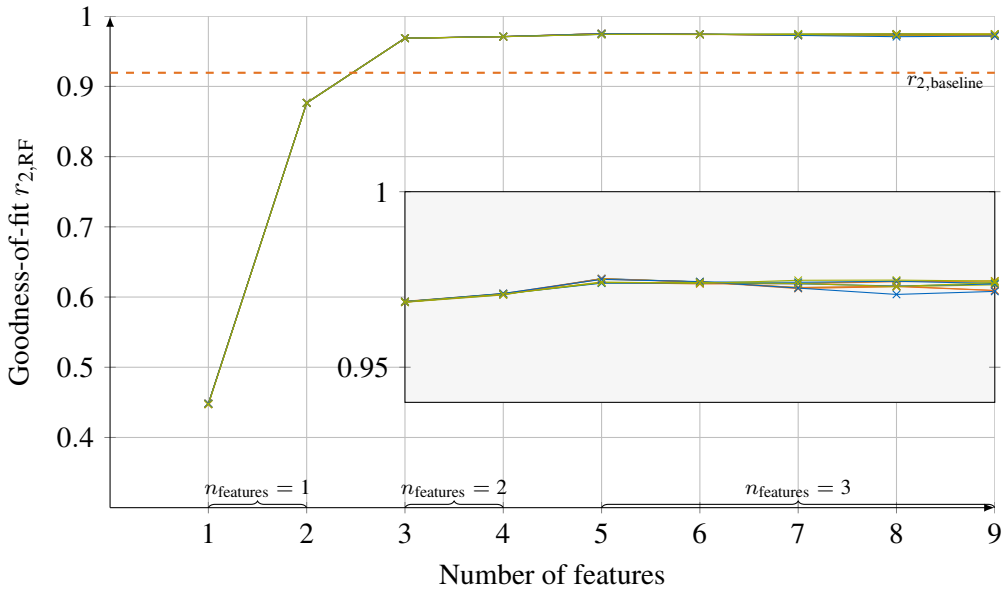


Figure 6.5: Goodness of fit in terms of the $r_{2,\mathrm{RF}}$ score (ref. (6.59)) of greedy feature selection using Random Forest Regression with hyper parameters given in Table 6.3b. The parameter n_{features} denotes the number of features considered for each split. Dashed orange line: baseline performance, see (6.65).

split as it offers a compromise between decorrelating the trees of the RF and exploiting the available features. The original paper [56] used $n_{\text{sqrt}} = \lfloor 1 + \log_2 \dim(\mathbf{z}) \rfloor$, where $\lfloor \bullet \rfloor$ rounds to the next smaller integer.

In Figure 6.5 we depict the $r_{2,\text{RF}}$ score over the number of selected features for ten runs of forward selection. As a baseline, we indicate the RF performance that is achieved by using the same hyper parameters without any feature extraction and selection, i.e., with the scene tuple as the feature vector, i.e.,

$$r_{2,\text{baseline}} = r_{2,\text{RF}}(\mathcal{D}_{\text{tr}}),\ \mathcal{D}_{\text{tr}} = \{\mathcal{S}_i, c_i\}. \tag{6.65}$$

Clearly, the performance of the RF criticality estimator exceeds the baseline with only three of the hand-crafted features. We can also see from Figure 6.5 that the $r_{2,\text{RF}}$ score for all feature selection runs is better than 0.95 and that the scores show a low variance between runs.

Selected features and parameters In order to arrive at a single best feature set, we chose the highest scoring feature set for each run of feature selection. These highest scoring feature sets are candidates for the best feature set. For each candidate, we perform a grid search (ref. Table (6.3b)) to individually determine the best set of hyper parameters. The final combination of features and hyper parameters is the one that yields the best generalization score.

In Tables 6.4a and 6.4b, we list the most informative set of features in the order they were selected by greedy feature selection as well as the hyper parameters found by grid search. The generalization score obtained using this feature-parameter combination is $r_{2,\text{RF}} = 0.976$. The first two selected features are the initial radial acceleration for lane keeping and the lateral avoidance acceleration. Since we use the MINA-LT OCP formulation for labeling, the criticality labels are proportional to the maximum acceleration norm. The radial acceleration is a reasonable first selection as it provides the minimum criticality for critical and uncritical driving scenarios. The lateral avoidance acceleration as the second feature is also well interpretable since we simulate avoidance maneuvers with significant lateral acceleration components. A simple, hand-crafted feature that was selected is the geometric ratio between the distance to the obstacle and the distance for lateral avoidance. The features relative distance, velocity, and acceleration features are highly indicative for the time-to-collision and the necessary avoidance accelerations. While this motivates their selection, it appears unintuitive that the time-to-collision itself was not chosen as a feature instead. A possible explanation is that the TTC and avoidance acceleration are already represented by the lateral avoidance acceleration feature. Thus, adding the TTC as a feature might only add redundancy and correlation among the features. Overall, the selected features appear to be reasonable choices and lead to a generalization performance that significantly exceeds the baseline performance of directly using the scene tuple as the

Table 6.4: Best features and hyper parameters for criticality regression with a generalization score of $r_{2,\mathrm{RF}} = 0.976$

(a) Best feature set in the order of forward selection

Selected feature	Variable, Formula
Radial acceleration	$z_1 = \frac{v_{\mathrm{h},0}^2}{c_0}$
Lateral avoidance acceleration	$z_2 = a_{\mathrm{y}}$, see (6.57)
Distance	$z_3 = \Delta x$, see 2.2
Geom. ratio: offset over distance	$z_4 = \frac{^{\mathrm{C}}y}{\Delta x}$
Object deceleration	$z_5 = a_{\mathrm{obs},0}$, see 2.2
Relative velocity	$z_6 = \Delta v$, see 2.2

(b) Best hyper parameters found with grid search

Hyper parameter	Variable	Value
Ensemble size	n_{trees}	1500
Features per split	n_{features}	4
Min. nb. of samples per split	$n_{\mathrm{split,min}}$	2
Max. tree depth	d_{max}	None
Split criterion		MSE

feature vector.

The greedy forward search described here is only one suboptimal way of feature selection, see, e.g., [127] for an overview. A popular alternative to greedy forward search is recursive feature elimination using Random Forests. There, starting from a large set of features, the RF is used to compute feature importance scores, i.e., the contribution of each feature to the generalization performance of the RF. The feature importance is used to prune features from the feature set that do not contribute to the RF performance. One caveat of such a recursive elimination approach is that it requires the RF to attain a rather good fit to the data in order for the feature importance to be meaningful and reliable. Indeed, the recursive elimination approach was not successful for this application since the RF does not fit well do the data if given all candidate features at once. An advanced combination of forward selection with feature elimination is described in [57].

Chapter 7

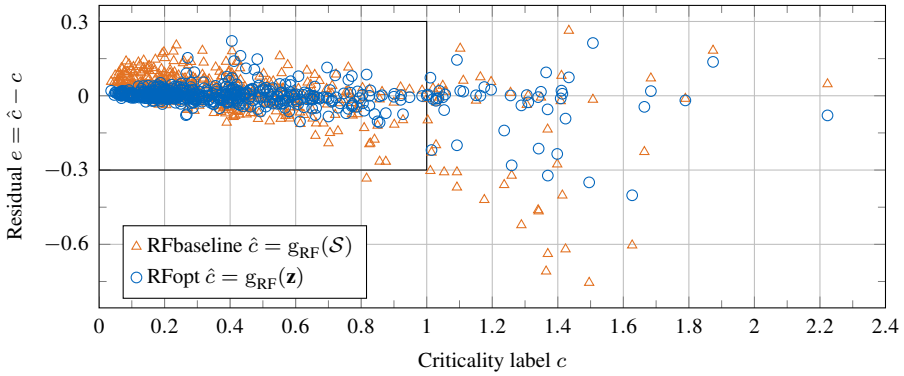
Validation of Criticality Labeling and Regression

In this section, we evaluate the performance of the criticality estimation based on Random Forest regression described in the previous Chapter. First, we quantify the generalization error of the estimator on a validation dataset, which is the standard procedure of evaluating a regression model in machine learning. Second, we evaluate the prediction performance in hand-crafted scenarios, looking at the criticality estimate compared to the label as the scenario evolves over time. Finally, we provide resimulation results for driving scene recorded on a test track with a moving obstacle vehicle on a bent road.

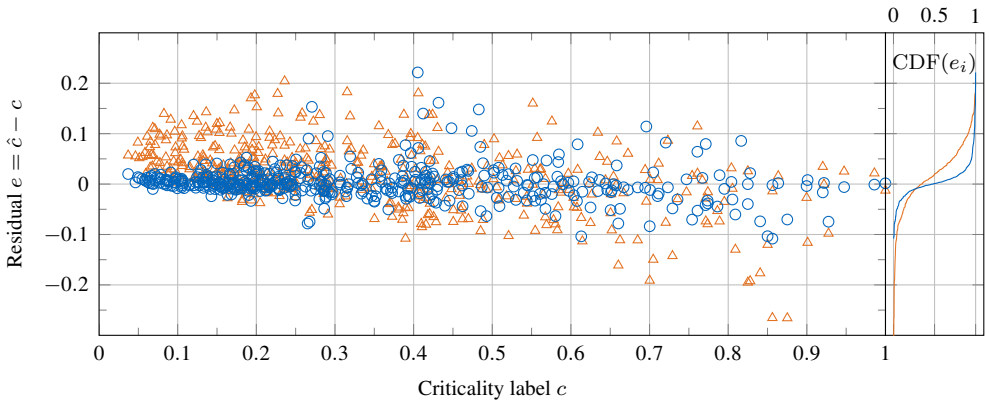
The criticality labels used throughout this chapter are based on the solutions of the min-max acceleration OCP MINA-LT, as defined in 4.47. The training data is generated using the stochastic scene sampling scheme described in Chapter 6.2, while feature vector $\mathbf{z}$ and the RF hyper parameters are given in Tables 6.4a and 6.4b, respectively. For the training and predictions of the statistical criticality estimator, we use the Random Forest implementation of the Python library *scikit-learn* [128].

7.1 Generalization Error

Prediction residuals on the validation data Figure 7.1a depicts the criticality labels c_i and the prediction error or *residuals* $e_i = \mathrm{g}_{\mathrm{RF}}(\mathbf{z}_i) - c_i$ for a random subset of five hundred predictions of the validation dataset, see Chapter 6.2.1. A goal of the stochastic model for scene sampling described in Chapter 6.2 is the even coverage of the entire criticality range. This goal is partially achieved. While the labels cover the entire relevant range of criticalities from zero to one, the label density decreases with



(a) Overview of the label distribution and residuals



(b) Samples $c_i \leq 1$, right: cumulative distribution of residuals CDF(e_i)

Figure 7.1: Residual plots of a random subset of five hundred criticality labels from the validation dataset

increasing criticality. This can be caused by the structure of the stochastic scene model itself, since it is not possible to directly sample the criticality value itself. Instead, uniform distributions on surrogate values like the time-to-collision were used. More importantly, scenarios with a high criticality might not be solvable by the OCP which can contribute to the lower density of labels in high criticality areas.

We can also recognize the increasing absolute values of the residuals with increasing criticality, particularly for the baseline RF. In Figure 7.1b, we focus on the most relevant range of criticalities from zero to one. The marginal distribution of the criticality labels c_i is given by the gray histogram and confirms the diminishing label density for high-criticality scenarios. For the baseline RF, a clear positive bias can be seen for low criticality scenarios, meaning that it tends to overestimate the threat. In contrast, the RF with feature selection shows no visible bias and retains a low prediction variance over the whole range of criticalities.

Generalization score The generalization score of the Random Forest criticality estimator with feature selection is $r_2 = 0.986$ (see (6.58)). This score is slightly higher than the score of $r_{2,\text{RF}} = 0.976$ predicted by out-of-bag estimation is Chapter 6.4 (see Table 6.4a and Figure 6.5). On one hand, this observation can have a random component, since such summary statistics are random variables themselves. On the other hand, out-of-bag estimates are known [56, Ch. 3] to underestimate the generalization score since they only measure the performance of an ensemble of base learners that is trained on two-thirds of the training data. Therefore, it is intuitive that the Random Forest achieves a better generalization if it is trained on the entire training set.

The overall impression from the residual plots and generalization score is that the Random Forest with feature extraction and selection achieves a significant increase in the quality of criticality prediction compared to directly using scene tuples. With a score of $r_2 = 0.986$, the resulting criticality estimator generalizes reliably to unseen data if that data is drawn from the same distribution.

One Random Forest prediction of a label in the validation dataset takes an average of 300μs on an Intel core i7-3740QM processor. Thus, the RF criticality estimator is clearly fast enough for use in real time situation interpretation algorithms. An important drawback of the RF is its large model size since the decision trees of the RF ensemble are unpruned [67]. If necessary for hard real-time application, the number of comparisons needed for the RF prediction as well as the model size can be upper bounded by limiting the number of trees and tree depth during the hyper parameter optimization.

7.2 Evaluation on Synthetic Collision Scenes

In addition to the generalization score reported in the previous section, we can gain insights into the criticality labeling and prediction technique by inspecting a range of driving scenarios on a case-by-case basis. To this end, we simulate the criticality of collision scenes on a grid of host vehicle velocities and obstacle velocities on a straight road segment.

In each simulation, the host vehicle front and the rear end of the obstacle are placed at an initial distance that corresponds to a time-to-collision of 2.4s. The velocities of the host vehicle and of the obstacle are constant throughout the simulations. The lateral position of the host vehicle with respect to the lane marking is ${}^{\mathrm{C}}y_0 = -2\mathrm{m}$ and it is assumed that the obstacle blocks the entire lane it occupies, see Figure 2.2. Both lanes have a width of $B_{\mathrm{r}} = B_{\mathrm{l}} = 4m$. The simulation ends at the time $t = 0$, the moment the host vehicle front intersects with the rear end of the obstacle.

For each point in time t_n, we generate the corresponding scene tuple $\mathcal{S}[n]$. Given the scene tuple, we compute the reference label $c[n] = \mathrm{c}(\mathcal{S}[n])$ by solving the criticality labeling OCP MINA-LT, see Table 4.1. We then compare the reference labels to the criticality predictions $\hat{c}[n] = \mathrm{g}_{\mathrm{RF}}(\mathbf{z}[n])$ using feature vectors $\mathbf{z}[n]$ as defined in Table 6.4a. As before, a Random Forest trained on the scene tuples is used as a baseline.

Figure 7.2 depicts the labels and predictions over time for combinations of initial host vehicle velocities $v_{\mathrm{h},0} \in \{10\mathrm{ms}^{-1}, 20\mathrm{ms}^{-1}, 30\mathrm{ms}^{-1}\}$ and obstacle velocities $v_{\mathrm{obs},0} \in \{0, \frac{1}{3}v_{\mathrm{h},0}, \frac{2}{3}v_{\mathrm{h},0}\}$.

Over almost the entire grid of velocities, the Random Forest estimator with feature extraction $\mathrm{g}_{\mathrm{RF}}(\mathbf{z})$ achieves a very close fit to the criticality labels determined by optimal control. One exception is shown in the top-right plot ($v_{\mathrm{h},0} = 10\mathrm{ms}^{-1}, v_{\mathrm{obs},0} = 6.66\mathrm{ms}^{-1}$), where the estimator shows a positive bias. The performance of the baseline estimator without feature extraction reaffirms the necessity of judicious feature engineering.

For all considered scenarios, the true criticality c lies within the ensemble percentiles $\hat{c}_5, \hat{c}_{95}$. Furthermore, it is evident that the difference $\hat{c}_{95} - \hat{c}_5$ correctly predicts the adequacy of the criticality estimation of the Random Forest, which can be used for a kind of self-diagnosis of a situation interpretation module. In the top-right scenario, the percentile difference correctly indicates a low prediction accuracy.

The center right plot illustrates a drawback of the optimal control labeling technique. Although the scenario changes smoothly with time, the criticality label has noticeable local jump as highlighted in the plot. One possible explanation is the fact that the optimal control problem (5.7) is non-convex such that the numerical optimization can only find local optima of the cost function. The outlier is likely to be caused by the numerical solver converging to a local minimum or saddle point of the

criticality cost function. Also note that the size of the interval $\hat{c}_{95} - \hat{c}_5$ is increased right where the label exhibits this instability. Since the Random Forest approximates the behavior of the labeling OCP, the instability of the label at that point of the feature space is reflected in the large spread of the ensemble predictions of the RF. Conversely, it might be possible to detect non-optimal labels from the training dataset with the goal of improving the label quality. The labeling procedure could be made more robust against local optima, e.g., by employing global optimization techniques like multi-start optimization with a diverse or even randomized set of initial guesses for the initialization of the numerical optimization solver[1].

7.3 Resimulation of a Dynamic Avoidance Maneuver

During the development and testing of active safety for series production cars, an enormous amount of sensor data is recorded and archived, stemming from proving grounds and public roads. An important reason for archiving this data is to comply with laws and regulations, which require thorough testing and documentation of safety critical systems. We can also use these recordings as test data to evaluate the performance of new systems and safety functions in a cost and time effective way without having to prepare a test vehicle. We refer to this approach to performance testing as *resimulation*. Note that resimulation is necessarily open-loop, which means that we cannot influence the state of the host vehicle, e.g., through braking or steering. Nevertheless, we can evaluate the activation times of the criticality interventions. At the same time, we can analyze - albeit only subjectively - the recorded video of the driving scene to judge the adequacy of our algorithms. Applied to criticality estimation, resimulation allows us to show that the proposed functionality is applicable to current and future series production cars and sensor system or - in the negative outcome - to find properties of current sensor systems or of the proposed functions that would make the real-world implementation challenging or impossible.

7.3.1 Experimental Setup and Data Processing

Our test drives were recorded on a prototype of the 2016 AUDI Q7 Sports Utility Vehicle equipped with a mono video camera for lane marking estimation and obstacle detection. We obtain the host vehicle state, lane markings, and information about the state of obstacles through the FlexRay protocol of the vehicle communication bus, using series production sensors. During the test drives, the test vehicle is equipped with a differential GPS system and high-precision gyroscopes, which send their state estimates over the CAN bus. These are used to determine the position of the host

[1] Refer to Section 5.2 for the initialization technique.

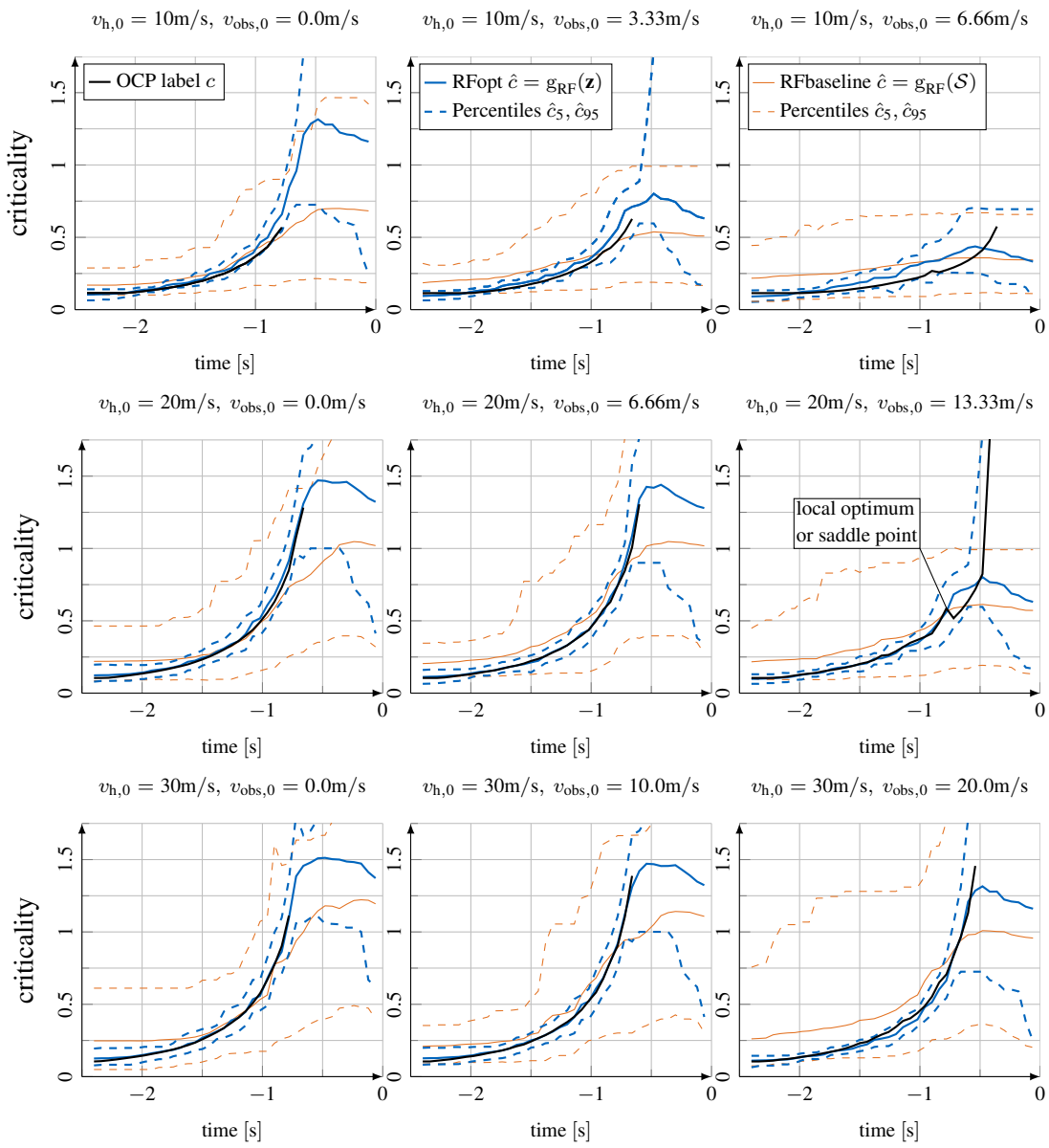


Figure 7.2: Criticality labels, predictions, and ensemble percentiles for collision scenes with varying EGO and obstacle velocities on a straight road with width $B_r = B_l = 4\text{m}$ and initial lateral offset ${}^{C}y_0 = -2\text{m}$

vehicle and obstacle in an inertial coordinate system, which allows us to visualize the movement over time.

During resimulation, the time series data available as CAN and FlexRay recordings is replayed and serves as the input for scene interpretation. In order to use this real-world sensor data for the evaluation of our criticality labeling and prediction approach, we must map the data to our scene representation, which involves multiple steps, i.e., data quality assurance, time-series resampling, coordinate-transformation, and scene extraction.

Data quality The first step that is necessary before we can label or estimate the criticality of a scene is to make sure that all relevant signals for scene interpretation are available with sufficient certainty. Lane markings, for example, cannot always be detected by a video camera since they might not be present on the road or they could be occluded by other vehicles or obstacles. To this end, quality indicators like the likelihood of existence of a lane marking or standard deviations are available. We remove time stamps from the resimulation where the likelihood of existence is below 0.8, which was determined to be an acceptable level of certainty by trial and error. Those time stamps are then replaced by interpolation between adjacent time stamps. In practice, it would be necessary to gracefully reduce the level of detail considered for criticality labeling by removing constraints that depend on signals that are not available in sufficient quality.

Time-series resampling Data sources like video, radar, and inertial sensors are physically distributed throughout the vehicle and are integrated on different electronic control units. Thus, they generally operate asynchronously, i.e., their signals are not updated at same time or at the same rate. By resampling, we extract a time-series vector of the necessary measurements with a common cycle time of $T = 100$ms.

In the following, we neglect the effects of jitter, i.e., fluctuations of the arrival time of a signal, and signals that are not updated periodically. We use the index $j \in 1, \ldots, N_v$ to distinguish between N_v signal sources, each with their own cycle time and offset. The j-th signal available on the bus system is updated at the time stamps

$$s_j[m] = s_j(t_m), \quad t_m = \tau_j + mT_j,\ m \in 0, \ldots, N_s - 1, \tag{7.1}$$

where τ_j is the offset and T_j is the cycle time of the j-th signal.

The goal of the time series resampling step is to extract a signal vector,

$$\mathbf{v}[n] = \begin{bmatrix} v_1(nT) & \ldots & v_{N_v}(nT) \end{bmatrix}^{\mathrm{T}} \in \mathbb{R}^{N_v}, \quad t_n = nT,\ n \in \mathbb{N} \tag{7.2}$$

that contains all N_v signal sources and uses a common cycle time T. To obtain the entries $v_j[n]$, we have to interpolate the values of the signals $s_j[m]$ at the time stamps t_n. A simple and very common way of interpolation is known as *sample-and-hold*, where $v_j[n]$ is equal to the last value of the signal s_j before the time nT. During the runtime of the vehicle, we only have access to past samples of the signals, so sample-and-hold can be a fine choice. Formally, the index of the last update $s_j[m_{j,n}]$ is given by

$$m_{j,n} = \left\lfloor \frac{t_n - \tau_j}{T_j} \right\rfloor, \tag{7.3}$$

where $\lfloor x \rfloor$ maps a real number x to the largest integer smaller than x. The resampled signal is given by

$$v_j[n] = v_j(nT) = s_j[m_{j,n}], \tag{7.4}$$

In resimulation, we have access to the complete time series $s_j[m], m \in 0, \ldots, N_j - 1$ sampled at times $t_{j,m}$. Therefore, we can use linear interpolation between $s_j[m_{j,n}]$ and $s_j[m_{j,n} + 1]$ to derive $\mathbf{v}[n]$, i.e.,

$$v_j[n] = s_j[m_{j,n}] + (nT - t_{j,m_{j,n}}) \frac{s_j[m_{j,n} + 1] - s_j[m_{j,n}]}{T_j}, \tag{7.5}$$

with $m_{j,n}$ given in (7.4). In our experiment, we used linear interpolation to replace a signal $s_j[m]$ if its variance was too high, see above.

Extraction of the scene model from BUS data The environment perception as well as the positioning data of the host vehicle is available on its BUS system. This information is provided in different coordinate systems, e.g., in ellipsoidal GPS coordinates and in the host vehicle coordinate system. Before the environment perception can be labeled by optimal control, the scene tuple representation $\mathcal{S}$ has to be extracted. Details on the required data processing steps are provided in Appendix A.

Labeling and prediction After extracting the scene tuples $\mathcal{S}[n]$ from the resampled signals $\mathbf{v}[n]$, we label the criticality values $c[n]$ by solving the collision avoidance OCP (5.7). The criticality estimates are computed by extracting the feature vectors $\mathbf{z}[n]$ from the scene tuples and using them as inputs for the regression and classification estimators.

7.3.2 Results of Resimulation

We consider a dynamic overtaking maneuver where the host vehicle keeps a velocity of around 70kmh^{-1} while closing in on a vehicle with a speed of 40kmh^{-1} on a

curved road. Shortly before a collision could occur, the host vehicle rapidly swerves to the left to avoid a collision. The first row in Figure 7.3 contains snapshots of the mono camera image at $t_1 = 8\text{s},\ t_2 = 9\text{s},\ t_3 = 9.7\text{s}$. In all four snapshots, we fix the origin of the Cartesian coordinate system to the position of the host vehicle in the first snapshot and keep a trace of past lane markings for orientation. The second row shows the scene representation according to the environment model defined in Chapter 2, reconstructed from the sensor measurements. The criticality plot in the third row shows the criticality as labeled by solving the numerical OCP (5.7), as well as the real-valued criticality estimation obtained from Random Forest regression. The reliability of the predictions is indicated by the 5 and 95 percentiles of the tree predictions.

From the criticality plots it is evident that in the considered test case, the regression estimate is a very close approximation to the reference label c. This confirms the performance of the Random Forest criticality estimator observed in the residual plot of Figure 7.1b and case-by-case evaluation in Figure 7.2.

Note that the label c itself is not available past $t = 9.5$, i.e., the numerical optimization did not find a feasible avoidance trajectory. This observation is at odds with the fact that we know from the recording that the collision is avoidable - although only by an intense swerving maneuver. It is caused by the fact that our OCP formulations assume that the host vehicle follows the lane parallel to the center marking. This is appropriate for an assistance system that protects inattentive drivers. After about $t = 9.3\text{s}$, this assumption no longer matches the dynamics of the real-world vehicle. Specifically, the real vehicle continuously increases the yaw angle, yaw rate, and steering angle during the start of the avoidance maneuver which builds up the model mismatch. In consequence, the avoidance trajectory in the OCP simulation would have to use a significantly higher lateral acceleration to avoid the collision. Since the OCP takes into account upper and lower bound constraints on the steering rate and braking force, the numerical solver is unable to find a feasible solution. Future work should therefore expand the optimal control formulation for criticality labeling to incorporate more dynamic initial state constraints.

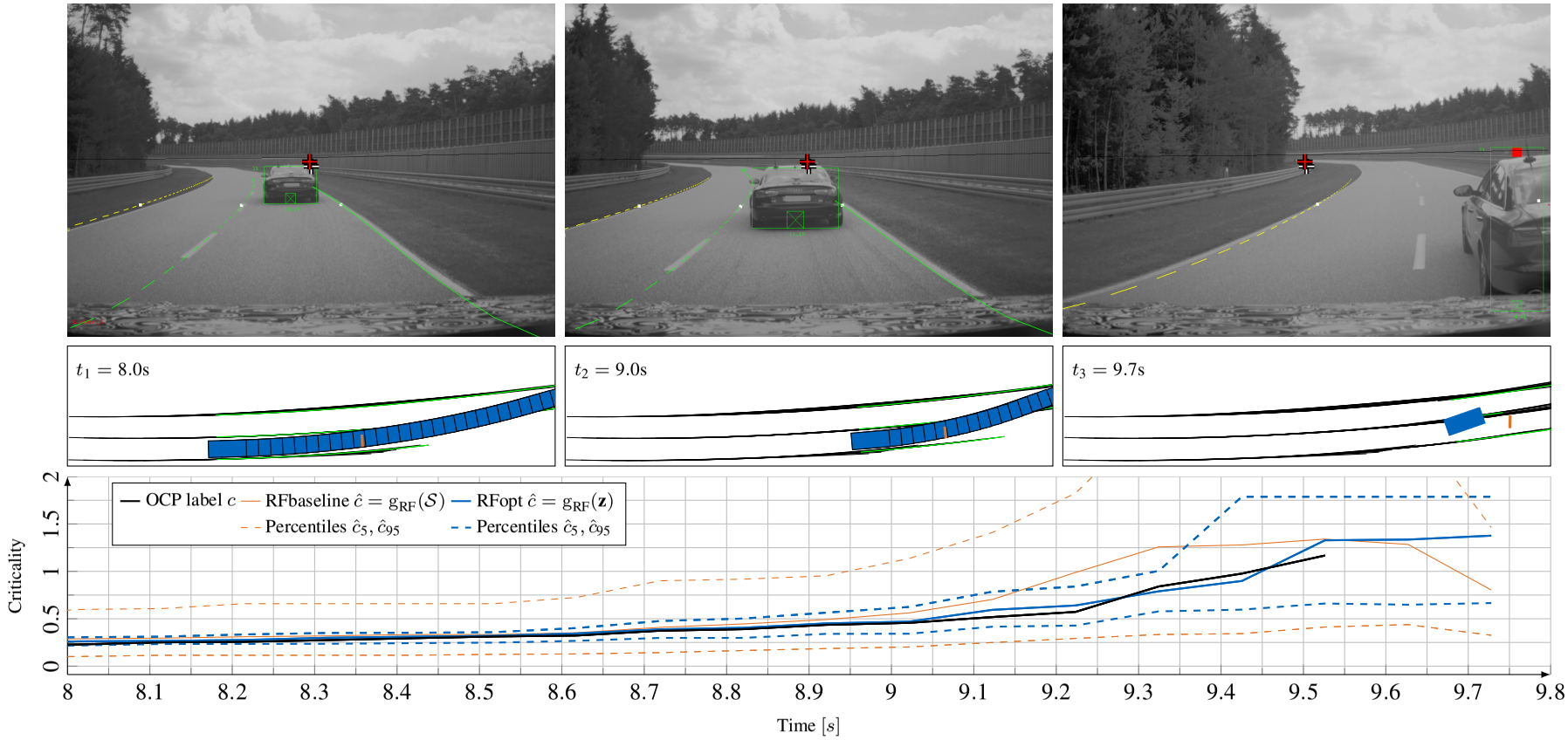


Figure 7.3: Criticality labels and estimates on a test track with a moving obstacle vehicle

Chapter 8

Conclusion and Outlook

Situation interpretation algorithms for active safety and automatic driving systems require a measure of the criticality or threat of the current driving situation to plan and trigger collision avoidance maneuvers. In this work, we have focused on the ground-truth labeling and the prediction of the criticality of automatic collision avoidance maneuvers with combined braking and steering in rear-end collision scenarios.

In the first part of this work, we have posed the criticality labeling problem in terms of a trajectory optimization using nonlinear optimal control. As the cost function, we have proposed to use the minimal maximum-norm of the acceleration vector during a prediction horizon of a few seconds. This value measures the peak intensity of the least intense avoidance trajectory [46]. Directly optimizing the criticality measure allows us to interpret the optimal values found by numerical optimization as a lower bound of the threat of the driving scenario. We have improved upon previous work by [46, 48] by formulating lane keeping constraints for general curvilinear lane boundaries and anti-collision constraints for polyhedral obstacles based on separating hyperplanes. Furthermore, we have identified alternative combinations of cost functions and constraint formulations and characterized the tradeoff between the feasibility of the optimal control problem, its interpretability as a criticality value and the necessary level of model detail.

In the second part of this work, we have exploited the ability to generate ground-truth labels by developing a reliable criticality estimator based on Random Forest (RF) regression to predict the criticality label in a computationally efficient way. For the generation of training data for subsequent machine learning steps, we have introduced an a priori probability distribution for the stochastic generation of critical longitudinal collision scenarios on curved lane segments. We have generated and selected a set of features using greedy forward feature selection that improves the generalization

score r_2 of the Random Forest to over 98%. The quality of the feature vectors used for the training of the Random Forest has a significant role in the accuracy of the estimator which as shown by comparison to a baseline Random Forest that naively uses the scene tuples for training. Using a combination of statistical performance evaluation, simulations of collision scenarios, and the resimulation of a lateral collision avoidance maneuver on a test track, we have validated the combination of optimal control labeling and Random Forest regression for criticality estimation.

The high fidelity and execution speed of the RF criticality estimator makes it suitable for real-time operation on an electronic control unit. It could also be paired with model-based algorithms in safety-critical systems that require a diversity and redundancy of algorithms and methodology [67]. Another application is the use of the RF estimator for data mining of critical — and therefore informative and challenging — driving scenes from large corpora of recorded driving data, e.g., from naturalistic driving studies or large-scale vehicle validation campaigns.

Since machine learning approaches to situation interpretation can only ever be as good or bad as the labeling process, they also inherit its inadequacies which points to opportunities for future work. First, the current formulation of the OCP is restricted to stationary lane following as an initial condition for trajectory optimization. It should be extended to additional initial state constraints. An important drawback of labeling with numerical optimal control is the fact that the numerical solver can get stuck in local optima, thus failing to converge to the global optimum of the optimal control problem. This leads to a kind of label fluctuation or label noise which increases the variance of the regression estimator as illustrated in Chapter 7.2. While, in our experience, these local optima are rare and only add a small perturbation to the label, future work on criticality labeling should try to control and reduce local optima, e.g., using multi-start optimization. The gold standard for the labeling of the desired behavior of active safety systems lies in globally optimal or epsilon-optimal control (see, e.g., [129]), which would avoid the issue of label noise altogether or provide a known upper bound on the sub-optimality.

Next, the criticality estimator could be trained to predict multiple aspects of criticality to provide a more nuanced situation interpretation, e.g., a combination of last-time-to-act, avoidance acceleration, and collision probability. Learning algorithms based on decision trees are trivially extensible to the multi-output case by just adding the additional information to the leaf nodes of each decision tree. More advanced techniques try and exploit correlations between these outputs to improve predictions, see, e.g., [130] for an overview of multiple-output regression.

Another line of inquiry would try to take into account sensor noise and the uncertainty of the behavior of other traffic participants. If a model of the sensor noise and obstacle behavior is available, it can be exploited during the training of a quantile regression forest [120]. The quantile regression forest could incorporate the sensor

noise and behavior uncertainty in the distribution of the ensemble predictions.

A vital direction for research on criticality regression is the exploitation of domain knowledge in the regression problem, to improve the prediction performance and to constrain the flexibility of the regression function. A step in this direction is made in [70] by imposing monotonicity constraints on the regression function, e.g., to enforce a monotone increase of the criticality value with increasing relative speed. During feature engineering, the practitioner is often looking for features that have a monotone relationship with the dependent variable, as seen in Chapter 6.3. Therefore, it appears that many such monotone relationships could be exploited.

The use of machine learning for safety-critical systems like active safety systems or autonomous driving is profoundly challenging from the perspective of functional safety and validation due to the black-box nature of many machine learning paradigms. A research direction that tries to pry open these black-boxes is referred to as *explainable* machine learning [131] which seeks to augment machine learning models with the capability to provide human-understandable explanations about how a prediction was reached. Efforts at the intersection of functional safety, validation, and machine learning combine exciting and challenging theory with an enormous potential impact on safety-critical industrial applications.

Appendix A

Extraction of Scene Data

During real-world testing of Active Safety Systems, it is often necessary to record the position and yaw angle of the host vehicle - and of any other vehicles or obstacles participating in experiments - in a common inertial coordinate system. This facilitates the resimulation and analysis of test drives by establishing a *ground-truth*, i.e., the best possible measurement and recording, that can be used to evaluate the performance of sensors and algorithms. In Table A.1, we list the BUS signals used for the resimulation of an avoidance maneuver. These signals are given in a variety of coordinate systems (CS), i.e., the EGO CS with its origin in the center of the rear axle, the camera CS, the road-fixed Cartesian coordinates of the so-called Local Tangent Plane (LTP), and the ellipsoidal coordinates of the WGS84 reference ellipsoid used in GPS navigation [132]. In the following, we extract the scene representation in terms of a scene tuple $\mathcal{S}$, ref. (2.12) from the signals in Table A.1.

Table A.1: BUS Signals used for resimulation and their coordinate systems

Signal	Symbol	Coord. System
Absolute Velocity	v	LTP
Yaw angle, rate	${}^{\mathrm{LTP}}\psi_{\mathrm{h}}$, ω	LTP
Curvature and rate of change of the curvature	c_0, κ	EGO
Origin of a lane marking	${}^{\mathrm{V}}\mathbf{o}$	EGO
Tangent angle at start of lane marking	${}^{\mathrm{V}}\psi$	EGO
Relative velocity components	${}^{\mathrm{V}}\Delta v_x$, ${}^{\mathrm{V}}\Delta v_y$	EGO
Radial distance, polar angle	r, ϕ	Camera
Latitude, longitude, and height	ϕ, λ, h	WGS84

Table A.2: Parameters of the WGS84 ellipsoid [132]

Parameter	Symbol	Value
Semi major axis	a	6378137m
Semi minor axis	b	6356752.3142m

A.1 (d)GPS and the Local Tangent Plane

Here, we briefly define the coordinate systems and transformations necessary for resimulating driving experiments that were recorded with a Differential Global Positioning System (dGPS). An in-depth exposition of (d)GPS can be found in [133] and [134]. dGPS improves the precision of GPS by using a local, stationary GPS receiver, that transmits correction signals to the receivers of all tracked objects. The correction signals contain information about the errors due to propagation delay, satellite clock error, and errors in the estimation of satellite positions (ephemeris) [133, Ch. 1], which is assumed to correlate for the reference station and the tracked objects. A dGPS system can achieve a precision of a few centimeters [134, Ch. 13]. The WGS84 ellipsoidal coordinates are supplied by a differential GPS. Furthermore, the yaw angle and yaw rates are measured using an inertial navigation system attached to the host vehicle. The yaw angle, which is measured by a gyroscope, is given in the so-called Local Tangent Plane (LTP), which is an inertial coordinate system (CS) which has its origin in the general area of the test site.

Ellipsoidal Coordinates of the World Geodetic System In GPS systems, the mean sea level of the earth is approximated by the WGS84 reference ellipsoid of revolution as defined by the international standard World Geodetic System [132]. This ellipsoid is centered at the earths center of mass and its rotational axis aligns with the rotational axis of the earth. The semi-axes of the ellipsoid are given in Table A.2.

The position of a terrestrial point of interest $\mathbf{p}(X, Y, Z)$ in this ellipsoidal reference system is given by the geodetic latitude ϕ, longitude λ, and height h, from which we can obtain the Cartesian coordinates X, Y, Z as [134]

$$\begin{aligned} X &= (R_N(\phi) + h)\cos(\phi)\cos(\lambda) \\ Y &= (R_N(\phi) + h)\cos(\phi)\sin(\lambda) \\ Z &= \left(\frac{b^2}{a^2}R_N(\phi) + h\right)\sin(\phi). \end{aligned} \tag{A.1}$$

Here, $R_N(\phi)$ is the radius of curvature of the prime vertical, see Figure A.1), which is

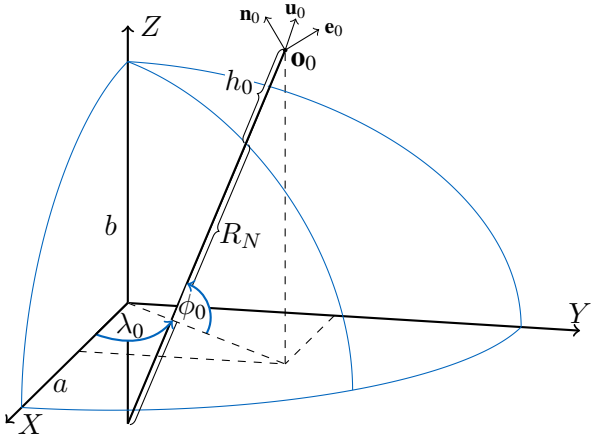


Figure A.1: WGS84 ellipsoidal reference system for terrestrial positioning and unit basis of a local tangent plane at $\mathbf{o}_0$

given by [134]

$$R_N = \frac{a^2}{\left(a^2 \cos^2(\phi_0) + b^2 \sin^2(\phi_0)\right)^{\frac{1}{2}}}. \tag{A.2}$$

Local Tangent Plane Since we use a Cartesian coordinate system (CS) attached to the road surface, the vehicle and object positions recorded with dGPS in terms of geodetic latitude, longitude and height, must be transformed to a local, road-fixed Cartesian coordinate system. This coordinate system is referred to as the local tangent plane (LTP).

A suitable choice for the origin of this local CS is the position of the reference dGPS receiver at the test site. We denote this origin as $\mathbf{o}_0$. The LTP is given by the basis vector pointing north, $\mathbf{n}_0 = \mathbf{n}(\phi_0, \lambda_0)$, and the basis vector pointing east, $\mathbf{e}_0 = \mathbf{e}(\phi_0, \lambda_0)$. These are often complemented with the normal vector $\mathbf{u}_0 = \mathbf{u}(\phi_0, \lambda_0)$ to form a north-east-up basis which is given by

$$\begin{bmatrix} \mathbf{n}_0 & \mathbf{e}_0 & \mathbf{u}_0 \end{bmatrix} = \begin{bmatrix} -\sin(\phi_0)\cos(\lambda_0) & -\sin(\lambda_0) & \cos(\phi_0)\cos(\lambda_0) \\ -\sin(\phi_0)\sin(\lambda_0) & \cos(\lambda_0) & \cos(\phi_0)\sin(\lambda_0) \\ \cos(\phi_0) & 0 & \sin(\phi_0) \end{bmatrix}. \tag{A.3}$$

We can now assign local Cartesian coordinates to all tracked objects $\mathbf{p}_i$ with respect to $\mathbf{o}_0$ by computing the projections

$${}^{\text{LTP}}\mathbf{p}_i = \begin{bmatrix} \mathbf{n}_0 & \mathbf{e}_0 \end{bmatrix}^{\mathrm{T}} (\mathbf{p}_i - \mathbf{o}_0). \tag{A.4}$$

In our experiments, we use a linear approximation to compute the coordinates of a point on the LTP by considering the radii of curvature at the point of reference $\mathbf{o}_0$. Movement along a line of constant latitude, i.e., movement in east-west direction, corresponds to rotation around the axis of revolution of the ellipse with a radius of $R_N(\phi)\cos(\lambda)$ [133]. The radius of curvature of a line of constant longitude (north-south), the *meridional* radius R_M, is given by [135, Ch.3.6]

$$R_M(\phi_0) = \frac{a(1-e^2)}{\left(1-e^2\sin^2(\phi_0)\right)^{\frac{3}{2}}}, \tag{A.5}$$

where e^2 is called the squared eccentricity of the ellipse,

$$e^2 = \frac{a^2-b^2}{a^2}. \tag{A.6}$$

With these radii of curvature, we can give the differentials in northern and eastern direction in the tangent plane, dN and dE, around the point of reference at (ϕ_0, λ_0, h_0) as

$$dE = (R_N(\phi_0)+h_0)\cos(\phi_0)d\lambda, \quad dN = (R_M(\phi_0)+h_0)d\phi. \tag{A.7}$$

Thus, we obtain a linear approximation for the two-dimensional LTP coordinates, which are increasing north and east, as

$$\begin{bmatrix} N \\ E \end{bmatrix} = \begin{bmatrix} (R_N(\phi_0)+h_0)\cos(\phi_0)(\phi-\phi_0) \\ (R_M(\phi_0)+h_0)(\lambda-\lambda_0) \end{bmatrix}. \tag{A.8}$$

We use the coordinates $(N,\ E)$ for positioning the EGO vehicle as well as the obstacle in the LTP around the dGSP base station.

From EGO coordinates to the LTP The positions of points ${}^{\mathrm{V}}\mathbf{p}$ detected by sensors like radar and video camera are given in the EGO coordinate system ${}^{\mathrm{V}}$ as defined in Section 2.3[1]. In order to compare these measurements to ground-truth positions, they must be expressed in the local tangent plane. Using the LTP coordinates of the host vehicle ${}^{\mathrm{LTP}}\mathbf{p}_{\mathrm{h}}$ obtained by dGPS, together with a yaw angle ${}^{\mathrm{LTP}}\psi_{\mathrm{h}}$ derived from gyroscope measurements, the transformation of a point $\mathbf{p}$ from EGO to LTP coordinates writes as

$${}^{\mathrm{LTP}}\mathbf{p} = \mathbf{R}\left({}^{\mathrm{LTP}}\psi_{\mathrm{h}}\right){}^{\mathrm{V}}\mathbf{p} + {}^{\mathrm{LTP}}\mathbf{p}_{\mathrm{h}}, \tag{A.9}$$

where $\mathbf{R}\left({}^{\mathrm{LTP}}\psi_{\mathrm{h}}\right)$ is a rotation matrix.

[1] On the vehicle BUS, the coordinates of objects in the environment is typically given with respect to an origin above the front or rear axle of the EGO vehicle. For simplicity, we assume that all data has already been transformed to the EGO CS ${}^{\mathrm{V}}$.

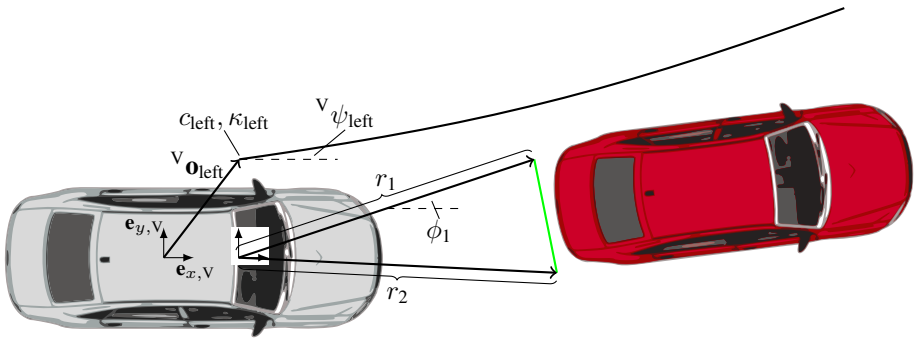


Figure A.2: EGO and camera coordinate systems, lane marking description and obstacle bounding box. Green: line between lower left and lower right corners of the obstacle bounding box.

A.2 Obstacles and Lane Markings

When an obstacle is detected in front of the host vehicle, it is assigned a bounding box that circumscribes the body of the obstacle as it appears in the camera plane. To construct the scene representation, we consider the positions of the bottom left and right corners of the bounding box to measure the width of the obstacle. These points are given in polar coordinates, where the radial distance r is measured from the position of the video camera to the point of the bounding box and the angular coordinate ϕ is given counter-clockwise starting from the longitudinal vehicle axis, see Figure A.2. Since we use a *flat world* scene interpretation, we do not make use of information relating to the height of the bounding box. The speed and acceleration of the obstacle are also given in camera coordinates.

With the video camera mounted at the position ${}^{\text{V}}\mathbf{r}_{\text{Cam}}$, the transformation from camera coordinates to EGO coordinates is given by

$$ {}^{\text{V}}\mathbf{r} = {}^{\text{V}}\mathbf{r}_{\text{Cam}} + \begin{bmatrix} r\cos(\phi) \\ r\sin(\phi) \end{bmatrix}. \tag{A.10} $$

The representation of segments of lane markings that are detected by the camera is depicted in Figure A.2. A lane marking is parameterized by its origin in EGO coordinates ${}^{\text{V}}\mathbf{o}_{\text{m}}$, the segment length Δl, as well as the yaw angle ${}^{\text{V}}\psi$ of the segments tangent vector at the origin, the curvature c at the origin, and the rate of change of the curvature κ, ref. Section 2.1.

Computing lane widths As in the scene model, we make the simplifying assumption of parallel lane markings. We only consider the case where we estimate the width B_{l} of the lane to the left of the EGO vehicle. Since the exact width of the lane occupied

by the obstacle does not influence the evasion maneuver, we assume equal lane widths, i.e., $B_\mathrm{r} = B_\mathrm{l}$. In general, we should always estimate the width of the lane in the direction of the considered evasion maneuver. For short distances, the clothoid path ${}^{\mathrm{V}}\mathbf{r}_{\mathrm{left}}(\lambda_{\mathrm{left}})$ of the left lane marking can be approximated by a straight line defined by its origin and tangent vectors, i.e.,

$$ {}^{\mathrm{V}}\mathbf{r}_{||}(\lambda_{\mathrm{left}}) = {}^{\mathrm{V}}\mathbf{o}_{\mathrm{left}} + {}^{\mathrm{V}}\mathbf{t}_{\mathrm{left}}\lambda_{\mathrm{left}},\ \lambda_{\mathrm{left}} \in \mathbb{R}. \tag{A.11} $$

Similarly, we have an orthogonal line in the direction of the normal vector of the center lane marking,

$$ {}^{\mathrm{V}}\mathbf{r}_{\perp}(\lambda_{\mathrm{normal}}) = {}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}} + {}^{\mathrm{V}}\mathbf{n}_{\mathrm{center}}\lambda_{\mathrm{normal}},\ \lambda_{\mathrm{normal}} \in \mathbb{R}. \tag{A.12} $$

In order to compute the lane width, we find the intersection of these lines:

$$ \begin{aligned} & {}^{\mathrm{V}}\mathbf{o}_{\mathrm{left}} + {}^{\mathrm{V}}\mathbf{t}_{\mathrm{left}}\lambda_{\mathrm{left}} = {}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}} + {}^{\mathrm{V}}\mathbf{n}_{\mathrm{center}}\lambda_{\mathrm{normal}}, \\ \Leftrightarrow \quad & {}^{\mathrm{V}}\mathbf{o}_{\mathrm{left}} - {}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}} = \begin{bmatrix} {}^{\mathrm{V}}-\mathbf{t}_{\mathrm{left}} & {}^{\mathrm{V}}\mathbf{n}_{\mathrm{center}} \end{bmatrix} \begin{bmatrix} \lambda_{\mathrm{left}} \\ \lambda_{\mathrm{normal}} \end{bmatrix}, \qquad (\mathrm{A.13}) \\ & \Leftrightarrow \begin{bmatrix} \lambda_{\mathrm{left}} \\ \lambda_{\mathrm{normal}} \end{bmatrix} = \begin{bmatrix} -{}^{\mathrm{V}}\mathbf{t}_{\mathrm{left}} & {}^{\mathrm{V}}\mathbf{n}_{\mathrm{center}} \end{bmatrix}^{-1} \left({}^{\mathrm{V}}\mathbf{o}_{\mathrm{left}} - {}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}} \right), \qquad (\mathrm{A.14}) \end{aligned} $$

which, under the realistic assumption that the normal and tangent vectors of the lane segments form an orthonormal basis, simplifies to

$$ \begin{bmatrix} \lambda_{\mathrm{left}} \\ \lambda_{\mathrm{normal}} \end{bmatrix} = \begin{bmatrix} -{}^{\mathrm{V}}\mathbf{t}_{\mathrm{left}} & {}^{\mathrm{V}}\mathbf{n}_{\mathrm{center}} \end{bmatrix}^{\mathrm{T}} \left({}^{\mathrm{V}}\mathbf{o}_{\mathrm{left}} - {}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}} \right). \tag{A.15} $$

Since the lane width is equal to the distance of the origin ${}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}}$ to the intersection point, it is obtained by the absolute value

$$ B_\mathrm{l} = |\lambda_{\mathrm{normal}}|. \tag{A.16} $$

Having estimated the line width, the lane parameters for the scene model are given by the curvature and rate of change of the curvature of the center lane marking, as well as the width of the left lane B_l.

Transformation to clothoid coordinates Recall that the trajectory optimization problem is most conveniently expressed in the inertial coordinate system of a clothoid ${}^{\mathrm{I}}$, see Section 4.1 on lane keeping constraints. Therefore, the state information about the lane markings and obstacles, given in the EGO CS ${}^{\mathrm{V}}$, have to be transformed to the inertial CS ${}^{\mathrm{I}}$ as detailed in Chapter 2.1.

Let us now express the initial position of the EGO vehicle as well as its yaw angle in the inertial coordinate system ${}^{\mathrm{I}}$. Since we are given estimates of the curvature and

rate of change of the curvature of the line segment, we can compute the arc length coordinate of the origin of the segment using (2.1) as

$$^{\mathrm{C}}x_{\mathrm{center}} = \frac{c_{\mathrm{center}}}{\kappa_{\mathrm{center}}}. \tag{A.17}$$

The yaw angle of the center marking observed by the host vehicle is

$$^{\mathrm{V}}\psi_{\mathrm{center}} = {}^{\mathrm{I}}\tau_{\mathrm{center}} - {}^{\mathrm{I}}\psi_{\mathrm{h}}, \tag{A.18}$$

which gives us the yaw angle of the host vehicle in the inertial coordinate system, i.e.,

$$^{\mathrm{I}}\psi_{\mathrm{h}} = 0.5c_{\mathrm{center}}^2\kappa_{\mathrm{center}} - {}^{\mathrm{V}}\psi_{\mathrm{center}}, \tag{A.19}$$

where we have inserted Equation (2.3) for the clothoid tangent angle $^{\mathrm{I}}\tau_{\mathrm{center}}$.

The arc length along the clothoid between the EGO vehicle position and the origin of the center marking segment is approximately given by the tangential component of the position of the origin, i.e.,

$$^{\mathrm{C}}\Delta x = {}^{\mathrm{C}}x_{\mathrm{center}} - {}^{\mathrm{C}}x_{\mathrm{h}} \approx {}^{\mathrm{V}}\mathbf{t}_{\mathrm{center}}^{\mathrm{T}}\,{}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}}, \tag{A.20}$$

which yields the arc length coordinate of the EGO in curvilinear coordinate system of the clothoid, i.e.,

$$^{\mathrm{C}}x_{\mathrm{h}} = {}^{\mathrm{C}}x_{\mathrm{center}} - {}^{\mathrm{V}}\mathbf{t}_{\mathrm{center}}^{\mathrm{T}}\,{}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}}. \tag{A.21}$$

Similarly, the offset coordinate of the host vehicle $^{\mathrm{C}}y$ is approximated by the normal component of the origin of the lane segment, i.e.,

$$^{\mathrm{C}}y_{\mathrm{h}} \approx {}^{\mathrm{V}}\mathbf{n}_{\mathrm{center}}\,{}^{\mathrm{V}}\mathbf{o}_{\mathrm{center}}. \tag{A.22}$$

To summarize, the video camera estimates the point of origin and tangent angle in the EGO CS $^{\mathrm{V}}$, the curvature, and rate of change of the curvature of the lane markings. From these, we extract the parameters (ref., Section 2.2) of the scene model, i.e., the lane widths $B_{\mathrm{l}},\ B_{\mathrm{r}}$, the reference clothoid, and the aligned coordinates $^{\mathrm{C}}x_{\mathrm{h}},\ {}^{\mathrm{C}}y_{\mathrm{h}}$ of the host vehicle.

List of Symbols

$\mathbf{e}$	Unit Vector
$\mathbf{0}$	Zero vector
$\mathbf{I}$	Identity matrix
$\mathbb{R}$	Set of real numbers
$\mathbb{N}$	Set of natural numbers
I	Superscript: road-fixed Cartesian coordinate system
C	Superscript: road-aligned coordinate system
V	Superscript: vehicle coordinate system
F	Superscript: Frenet coordinate system
LTP	Superscript: road-fixed Cartesian coordinate system in the local tangent plane
obs	Subscript: obstacle
h	Subscript: host/EGO vehicle
lon	Subscript: longitudinal direction
lat	Subscript: lateral direction
f	Subscript: front
r	Subscript: rear
μ	Friction coefficient
g	Gravitational acceleration on earth
m	Mass
β	Body slip angle
α	Tire slip angle
δ	Steering angle
ψ	Yaw angle
ω	Yaw rate
θ	Heading angle
v	Tangential velocity
a	Acceleration norm
k	Cornering stiffness
l_f	Distance of center of gravity from front axle
l_r	Distance of center of gravity from rear axle
I	Mass moment of inertia
$\mathbf{L}$	Angular momentum
$\mathbf{q}$	Torque vector
W	Vehicle width
L	Vehicle length
$\mathbf{s}$	State vector
u	Scalar control input

$\mathbf{u}$	Vector of control inputs
F	Tire force
k_{b}	Proportional braking factor
s	Subscript: sideways
t	Subscript: tangential
r	Polar radius, angular distance in camera coordinates
ϕ	Polar angle
$\mathcal{S}$	Scene tuple
B	Lane width
c	Curvature
κ	Curvature change rate
$\mathbf{r}$	Point on clothoid curve
$\mathbf{n}$	Normal vector of a line or curve
$\mathbf{t}$	Tangent vector of a line or curve
τ	Clothoid angle
l	Arclength along clothoid curve
T	Prediction horizon
Δt	Step size
n	Time index
b	Hyper plane offset
ϕ	Hyper plane angle
c	Real-valued criticality label
$\mathcal{D}$	Data set
M	Number of simulated scenes
$\mathbf{z}$	Feature vector
S'	Number of high-level features
N_{L}	Ensemble size
ϕ	Geodetic Latitude
λ	Geodetic Longitude
AEB	Automatic Emergency Braking
AIS	Abbreviated Injury Scale
ASS	Active Safety System
CART	Classification And Regression Tree
COD	Coefficient Of Determination
COG	Center Of Gravity
CS	Coordinate System
DMS	Direct Multiple Shooting
DT	Decision Tree
EA	Evolutionary Algorithm
ECDF	Empirical Cumulative Distribution Function

ECU	Electronic Control Unit
IVP	Initial Value Problem
KKT	Karush-Kuhn-Tucker
LTP	Local Tangent Plane
OCP	Optimal Control Problem
ODE	Ordinary Differential Equation
OOB	Out-of-Bag
PDF	Probability Density Function
PDIP	Primal-Dual Interior-Point
PMP	Pontryagin's Maximum Principle
RF	Random Forest
RK4	Runge-Kutta fourth-order
RRT	Rapidly-exploring Random Tree
SAE	Society of Automotive Engineers
SHT	Supporting Hyperplane Theorem
SI	Situation Interpretation
TTC	Time-To-Collision

List of Figures

List of Tables

Bibliography

[1] J. B. Cicchino, "Effectiveness of forward collision warning and autonomous emergency braking systems in reducing front-to-rear crash rates," *Accident Analysis & Prevention*, vol. 99, no. Part A, pp. 142 – 152, 2017. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0001457516304006

[2] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1, June 2005, pp. 886–893 vol. 1.

[3] D. G. Lowe, "Object recognition from local scale-invariant features," in *Computer Vision, 1999. Proceedings of the seventh IEEE International Conference on*, vol. 2, 1999, pp. 1150–1157.

[4] C. Cortes and V. Vapnik, "Support-vector networks," *Machine Learning*, vol. 20, no. 3, pp. 273–297, 1995.

[5] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119 – 139, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S002200009791504X

[6] M. Enzweiler and D. M. Gavrila, "Monocular pedestrian detection: Survey and experiments," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 31, no. 12, pp. 2179–2195, Dec 2009.

[7] P. Dollar, C. Wojek, B. Schiele, and P. Perona, "Pedestrian detection: An evaluation of the state of the art," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 34, no. 4, pp. 743–761, April 2012.

[8] R. Benenson, M. Omran, J. Hosang, and B. Schiele, *Computer Vision - ECCV 2014 Workshops: Zurich, Switzerland, September 6-7 and 12, 2014, Proceedings, Part II*. Springer International Publishing, 2015, vol. 8926, ch. Ten Years of Pedestrian Detection, What Have We Learned?, pp. 613–627.

[9] S. Sivaraman and M. M. Trivedi, "Looking at vehicles on the road: A survey of vision-based vehicle detection, tracking, and behavior analysis," *IEEE Transactions on Intelligent Transportation Systems*, vol. 14, no. 4, pp. 1773–1795, Dec 2013.

[10] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 435–444, 2015.

[11] J. Freyer, L. Winkler, M. Warnecke, and G.-P. Duba, "A little bit more attentive: Audi active lane assist," *ATZ Worldwide*, vol. 10, pp. 40–44, 2010.

[12] R. Behringer, "Road recognition from multifocal vision," in *Intelligent Vehicles '94 Symposium, Proceedings of the*, Oct 1994, pp. 302–307.

[13] E. D. Dickmanns, R. Behringer, D. Dickmanns, T. Hildebrandt, M. Maurer, F. Thomanek, and J. Schiehlen, "The seeing passenger car 'VaMoRs-P'," in *Intelligent Vehicles '94 Symposium, Proceedings of the*, Oct 1994, pp. 68–73.

[14] Y. Zhou, R. Xu, X. Hu, and Q. Ye, "A robust lane detection and tracking method based on computer vision," *Measurement Science and Technology*, vol. 17, no. 4, p. 736, 2006.

[15] Z. Kim, "Robust lane detection and tracking in challenging scenarios," *IEEE Transactions on Intelligent Transportation Systems*, vol. 9, no. 1, pp. 16–26, March 2008.

[16] H. Loose, "Dreidimensionale Straßenmodelle für Fahrerassistenzsysteme auf Landstraßen," Ph.D. dissertation, Karlsruhe Institut für Technologie, 2012.

[17] S. Tokoro, K. Kuroda, A. Kawakubo, K. Fujita, and H. Fujinami, "Electronically scanned millimeter-wave radar for pre-crash safety and adaptive cruise control system," in *Intelligent Vehicles Symposium, 2003. Proceedings. IEEE*, June 2003, pp. 304–309.

[18] J. Hasch, E. Topak, R. Schnabel, T. Zwick, R. Weigel, and C. Waldschmidt, "Millimeter-wave technology for automotive radar sensors in the 77 ghz frequency band," *IEEE Transactions on Microwave Theory and Techniques*, vol. 60, no. 3, pp. 845–860, March 2012.

[19] J. Effertz, "Autonome Fahrzeugführung in urbaner Umgebung durch Kombination objekt- und kartenbasierter Umfeldmodelle," Ph.D. dissertation, Technischen Universität Carolo-Wilhelmina zu Braunschweig, 2009.

[20] M. E. Bouzouraa, "Belegungskartenbasierte umfeldwahrnehmung in kombination mit objektbasierten ansätzen für fahrerassistenzsysteme," Dissertation, Technische Universität München, München, 2012.

[21] D. Geronimo, A. M. Lopez, A. D. Sappa, and T. Graf, "Survey of pedestrian detection for advanced driver assistance systems," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, no. 7, pp. 1239–1258, July 2010.

[22] J. Levinson, J. Askeland, J. Becker, J. Dolson, D. Held, S. Kammel, J. Z. Kolter, D. Langer, O. Pink, V. Pratt, M. Sokolsky, G. Stanek, D. Stavens, A. Teichman, M. Werling, and S. Thrun, "Towards fully autonomous driving: Systems and algorithms," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, June 2011, pp. 163–168.

[23] S. A. R. F., V. Fremont, P. Bonnifait, and V. Cherfaoui, "An embedded multimodal system for object localization and tracking," *IEEE Intelligent Transportation Systems Magazine*, vol. 4, no. 4, pp. 42–53, Winter 2012.

[24] M. M. Trivedi, T. Gandhi, and J. McCall, "Looking-in and looking-out of a vehicle: Computer-vision-based enhanced vehicle safety," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 1, pp. 108–120, March 2007.

[25] S. Kaplan, M. A. Guvensan, A. G. Yavuz, and Y. Karalurt, "Driver behavior analysis for safe driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 6, pp. 3017–3032, Dec 2015.

[26] J. Jansson, "Collision avoidance theory: With application to automotive collision mitigation," Ph.D. dissertation, Linköping University, 2005.

[27] S. Lefèvre, D. Vasquez, and C. Laugier, "A survey on motion prediction and risk assessment for intelligent vehicles," *ROBOMECH Journal*, vol. 1, no. 1, pp. 1–14, 2014. [Online]. Available: http://dx.doi.org/10.1186/s40648-014-0001-z

[28] A. Vahidi and A. Eskandarian, "Research advances in intelligent collision avoidance and adaptive cruise control," *IEEE Transactions on Intelligent Transportation Systems*, vol. 4, no. 3, pp. 143–153, Sept 2003.

[29] K. Vogel, "A comparison of headway and time to collision as safety indicators," *Accident Analysis & Prevention*, vol. 35, no. 3, pp. 427 – 433, 2003. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0001457502000222

[30] J. Hillenbrand, A. Spieker, and K. Kroschel, "A multilevel collision mitigation approach – its situation assessment, decision making, and performance

tradeoffs," *IEEE Trans. Intell. Transp. Syst.*, vol. 7, no. 4, pp. 528–540, Dec 2006.

[31] A. Tamke, T. Dang, and G. Breuel, "A flexible method for criticality assessment in driver assistance systems," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, June 2011, pp. 697–702.

[32] J. Jansson, J. Johansson, and F. Gustafsson, "Decision making for collision avoidance systems," SAE Technical Paper, Tech. Rep., 2002.

[33] A. Eidehall and L. Petersson, "Statistical threat assessment for general road scenes using monte carlo sampling," *IEEE Transactions on intelligent transportation systems*, vol. 9, no. 1, pp. 137–147, 2008.

[34] J. Jansson and F. Gustafsson, "A framework and automotive application of collision avoidance decision making," *Automatica*, vol. 44, no. 9, pp. 2347 – 2351, 2008. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109808000617

[35] A. Berthelot, A. Tamke, T. Dang, and G. Breuel, "Stochastic situation assessment in advanced driver assistance system for complex multi-objects traffic situations," in *Intelligent Robots and Systems (IROS), 2012 IEEE/RSJ International Conference on*, Oct 2012, pp. 1180–1185.

[36] M. Althoff, O. Stursberg, and M. Buss, "Model-based probabilistic collision detection in autonomous driving," *IEEE Transactions on Intelligent Transportation Systems*, vol. 10, no. 2, pp. 299–310, June 2009.

[37] M. Althoff, "Reachability Analysis and its Application to the Safety Assessment of Autonomous Cars," Ph.D. dissertation, Technische Universität München, February 2010.

[38] E. Coeling, L. Jakobsson, H. Lind, and M. Lindman, "Collision warning with auto brake Ű a real-life safety perspective," in *Proc. 20st International Technical Conference on the Enhanced Safety of Vehicles*, no. 07-0450, 2007.

[39] M. Distner, M. Bengtsson, T. Broberg, and L. Jakobsson, "City safety – a system addressing rear-end collisions at low speeds," in *Proc. Int. Tech. Conf. Enhanced Safety Veh., ESV*, no. 09-0371, 2009.

[40] A. Eidehall, "Multi-target threat assessment for automotive applications," in *Intelligent Transportation Systems (ITSC), 2011 14th International IEEE Conference on*, Oct 2011, pp. 433–438.

[41] M. Brannstrom, E. Coelingh, and J. Sjoberg, "Model-based threat assessment for avoiding arbitrary vehicle collisions," *IEEE Trans. Intell. Transp. Syst.*, vol. 11, no. 3, pp. 658–669, Sept 2010.

[42] A. Eidehall and D. Madas, "Real time path planning for threat assessment and collision avoidance by steering," in *Intelligent Transportation Systems - (ITSC), 2013 16th International IEEE Conference on*, Oct 2013, pp. 916–921.

[43] *Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*, On-Road Automated Driving (Orad) Committee Std.

[44] H. G. Bock and K.-J. Plitt, "A multiple shooting algorithm for direct solution of optimal control problems," in *Proc. IFAC World Congr. Budapest*, Proceedings 9th IFAC World Congress Budapest. Pergamon Press, 1984, pp. 242–247. [Online]. Available: http://www.iwr.uni-heidelberg.de/groups/agbock/FILES/Bock1984.pdf

[45] P. Tournassoud, "A strategy for obstacle avoidance and its application to mulltirobot systems," in *Robotics and Automation. Proceedings. 1986 IEEE International Conference on*, vol. 3, Apr 1986, pp. 1224–1229.

[46] S. Karrenberg, "Zur Erkennung unvermeidbarer Kollisionen von Kraftfahrzeugen mit Hilfe von Stellvertretertrajektorien," Ph.D. dissertation, Technische Universität Braunschweig, 2008.

[47] S. J. Anderson, S. C. Peters, T. E. Pilutti, and K. Iagnemma, "An optimal-control-based framework for trajectory planning, threat assessment, and semi-autonomous control of passenger vehicles in hazard avoidance scenarios," *Int. J. Veh. Autonomous Syst.*, vol. 8, no. 2, pp. 190–216, Jan. 2010.

[48] S. C. Peters, "Optimal planning and control for hazard avoidance of front-wheel steered ground vehicles," Ph.D. dissertation, Massachusetts Institute of Technology, 2012.

[49] I. Xausa, R. Baier, M. Gerdts, M. Gonter, and C. Wegwerth, "Avoidance trajectories for driver assistance systems via solvers for optimal control problems," in *Int. Symp. Math. Theory Networks Syst.*, 2012.

[50] I. Xausa, "Verification of collision avoidance systems using optimal control and sensitivity analysis," Ph.D. dissertation, Fakultät für Luft- und Raumfahrttechnik der Universität der Bundeswehr München, 2015.

[51] K. Lee and H. Peng, “Evaluation of automotive forward collision warning and collision avoidance algorithms,” *Vehicle System Dynamics*, vol. 43, no. 10, pp. 735–751, 2005.

[52] M. Althoff, M. Koschi, and S. Manzinger, “Commonroad: Composable benchmarks for motion planning on roads,” in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 719–726.

[53] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, Nov 1998.

[54] J. Deng, W. Dong, R. Socher, L. J. Li, K. Li, and L. Fei-Fei, “ImageNet: A large-scale hierarchical image database,” in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, June 2009, pp. 248–255.

[55] L. Yang, J. H. Yang, E. Feron, and V. Kulkarni, “Development of a performance-based approach for a rear-end collision warning and avoidance system for automobiles,” in *IEEE IV2003 Intelligent Vehicles Symposium. Proceedings*, June 2003, pp. 316–321.

[56] L. Breiman, “Random forests,” *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[57] M. Botsch and J. A. Nossek, “Feature selection for change detection in multivariate time-series,” in *IEEE Symp. Computational Intell. Data Mining, CIDM*, March 2007, pp. 590–597.

[58] M. Katagiri, J. Pramudita, Y. Miyazaki, and S. Ujihashi, “Development of occupant injury prediction algorithms for advanced automatic collision notification by numerical crash reconstructions,” in *23rd Enhanced Safety of Vehicles Conference (ESV)*, 2013.

[59] A. Meier, M. Gonter, and R. Kruse, “Precrash classification of car accidents for improved occupant safety systems,” *Procedia Technology*, vol. 15, no. Supplement C, pp. 198 – 207, 2014, 2nd International Conference on System-Integrated Intelligence: Challenges for Product and Production Engineering. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S221201731400187X

[60] M. Müller, P. Nadarajan, M. Botsch, W. Utschick, D. Böhmländer, and S. Katzenbogen, “A statistical learning approach for estimating the reliability of crash severity predictions,” in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Nov 2016, pp. 2199–2206.

[61] S. M. Lavalle and J. J. Kuffner, Jr., "Rapidly-exploring random trees: Progress and prospects," in *Algorithmic and Computational Robotics: New Directions*, 2000, pp. 293–308.

[62] R. Lachner, M. H. Breitner, and H. J. Pesch, "Real-time computation of strategies of differential games with applications to collision avoidance," in *Variational Calculus, Optimal Control and Applications*, ser. International Series of Numerical Mathematics, W. H. Schmidt, K. Heier, L. Bittner, and R. Bulirsch, Eds. Birkhäuser Basel, 1998, vol. 124, pp. 281–290.

[63] P. Nadarajan, M. Botsch, and S. Sardina, "Predicted-occupancy grids for vehicle safety applications based on autoencoders and the random forest algorithm," in *2017 International Joint Conference on Neural Networks (IJCNN)*, May 2017, pp. 1244–1251.

[64] A. Chaulwar, M. Botsch, and W. Utschick, "A machine learning based biased-sampling approach for planning safe trajectories in complex, dynamic traffic-scenarios," in *2017 IEEE Intelligent Vehicles Symposium (IV)*, June 2017, pp. 297–303.

[65] M. Bojarski, D. D. Testa, D. Dworakowski, B. Firner, B. Flepp, P. Goyal, L. D. Jackel, M. Monfort, U. Muller, J. Zhang, X. Zhang, J. Zhao, and K. Zieba, "End to end learning for self-driving cars," *CoRR*, vol. abs/1604.07316, 2016. [Online]. Available: http://arxiv.org/abs/1604.07316

[66] P. Koopman and M. Wagner, "Challenges in autonomous vehicle testing and validation," *SAE International Journal of Transportation Safety*, vol. 4, no. 1, pp. 15–24, April 2016.

[67] M. Botsch, "Machine Learning Techniques for Time Series Classification," Ph.D. dissertation, Technische Universität München, 2009.

[68] S. Herrmann, W. Utschick, M. Botsch, and F. Keck, "Supervised learning via optimal control labeling for criticality classification in vehicle active safety," in *Intelligent Transportation Systems (ITSC), 2015 IEEE 18th International Conference on*, Sept 2015, pp. 2024–2031.

[69] S. Herrmann and W. Utschick, "Availability and interpretability of optimal control for criticality estimation in vehicle active safety," in *2016 Design, Automation Test in Europe Conference Exhibition (DATE)*, March 2016, pp. 415–420.

[70] M. Knödlseder, "Regression with monotonicity constraints," Bachelor's thesis, Technische Universität München, 2016.

[71] A. Richards and J. How, "Model predictive control of vehicle maneuvers with guaranteed completion time and robust feasibility," in *American Control Conference. Proceedings of the 2003*, vol. 5, June 2003, pp. 4034–4040 vol.5.

[72] G. Wolf and W. Pietzsch, *Straßenplanung*, 7th ed., ser. Werner-Ingenieur-Texte. Werner Verlag, 2005.

[73] A. Eidehall and F. Gustafsson, "Combined road prediction and target tracking in collision avoidance," in *IEEE Intelligent Vehicles Symposium, 2004*, June 2004, pp. 619–624.

[74] S. Nedevschi, R. Schmidt, T. Graf, R. Danescu, D. Frentiu, T. Marita, F. Oniga, and C. Pocol, "3D lane detection system based on stereovision," in *Proceedings. The 7th International IEEE Conference on Intelligent Transportation Systems*, Oct 2004, pp. 161–166.

[75] A. Scheuer and T. Fraichard, "Continuous-curvature path planning for car-like vehicles," in *Intelligent Robots and Systems, 1997. IROS '97., Proceedings of the 1997 IEEE/RSJ International Conference on*, vol. 2, Sep 1997, pp. 997–1003 vol.2.

[76] *Richtlinien für die Anlage von Straßen - Teil Linienführung (RAS-L)*, Forschungsgesellschaft für Straßen- und Verkehrswesen Std.

[77] W. Kühnel, *Differential Geometry: Curves - Surfaces - Manifolds*, 3rd ed., ser. Student mathematical library, B. Hunt, Ed. American Mathematical Society, 2015, vol. 77.

[78] A. Eidehall, J. Pohl, F. Gustafsson, and J. Ekmark, "Toward autonomous collision avoidance by steering," *IEEE Transactions on Intelligent Transportation Systems*, vol. 8, no. 1, pp. 84–94, March 2007.

[79] R. N. Jazar, *Vehicle Dynamics: Theory and Application*, 2nd ed. Springer-Verlag New York, 2014.

[80] D. Schramm, M. Hiller, and R. Bardini, *Modellbildung und Simulation der Dynamik von Kraftfahrzeugen*. Springer, 2013.

[81] L. Rade, B. Westergren, and P. Vachenauer, *Springers mathematische Formeln: Taschenbuch für Ingenieure, Naturwissenschaftler, Informatiker, Wirtschaftswissenschaftler*. Springer-Verlag Berlin Heidelberg, 2000.

[82] E. Bakker, L. Nyborg, and H. B. Pacejka, "Tyre modelling for use in vehicle dynamics studies," SAE Technical Paper, Tech. Rep., 1987.

[83] B. Johansson and M. Gafvert, "Untripped suv rollover detection and prevention," in *Decision and Control, 2004. CDC. 43rd IEEE Conference on*, vol. 5, Dec 2004, pp. 5461–5466 Vol.5.

[84] K. Yi, K. Hedrick, and S.-C. Lee, "Estimation of tire-road friction using observer based identifiers," *Vehicle System Dynamics*, vol. 31, no. 4, pp. 233–261, 1999.

[85] F. Gustafsson, "Slip-based tire-road friction estimation," *Automatica*, vol. 33, no. 6, pp. 1087 – 1099, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109897000034

[86] L. R. Ray, "Nonlinear tire force estimation and road friction identification: Simulation and experiments," *Automatica*, vol. 33, no. 10, pp. 1819 – 1833, 1997. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S0005109897000939

[87] P. Jonsson, J. Casselgren, and B. Thörnberg, "Road surface status classification using spectral analysis of NIR camera images," *IEEE Sensors Journal*, vol. 15, no. 3, pp. 1641–1656, March 2015.

[88] A. Pohl, R. Steindl, and L. Reindl, "The 'intelligent tire' utilizing passive saw sensors measurement of tire friction," *IEEE Transactions on Instrumentation and Measurement*, vol. 48, no. 6, pp. 1041–1046, Dec 1999.

[89] F. Holzmann, M. Bellino, R. Siegwart, and H. Bubb, "Predictive estimation of the road-tire friction coefficient," in *2006 IEEE Conference on Computer Aided Control System Design, 2006 IEEE International Conference on Control Applications, 2006 IEEE International Symposium on Intelligent Control*, Oct 2006, pp. 885–890.

[90] H. Pacejka and I. Besselink, *Tire and Vehicle Dynamics*, 3rd ed. Butterworth-Heinemann, 2012.

[91] J. Svendenius, "Tire modeling and friction estimation," Ph.D. dissertation, Lund University, 2007.

[92] M. Diehl, H. G. Bock, H. Diedam, and P.-B. Wieber, "Fast Direct Multiple Shooting Algorithms for Optimal Robot Control," in *Fast Motions in Biomechanics and Robotics*, Heidelberg, Germany, 2005. [Online]. Available: https://hal.inria.fr/inria-00390435

[93] M. Bollhöfer and V. Mehrmann, *Numerische Mathematik: Eine projektorientierte Einführung für Ingenieure, Mathematiker und Naturwissenschaftler*. Vieweg+Teubner Verlag, 2004.

[94] J. Andersson, J. Akesson, and M. Diehl, "CasADi: A symbolic package for automatic differentiation and optimal control," in *Recent Advances in Algorithmic Differentiation*, ser. Lecture Notes in Computational Science and Engineering, S. Forth, P. Hovland, E. Phipps, J. Utke, and A. Walther, Eds. Springer, 2012, vol. 87, pp. 297–307.

[95] A. Wächter and L. T. Biegler, "On the implementation of an interior-point filter line-search algorithm for large-scale nonlinear programming," *Math. Programming*, vol. 106, no. 1, pp. 25–57, 2006.

[96] P. E. Gill, W. Murray, and M. A. Saunders, "SNOPT: An SQP algorithm for large-scale constrained optimization," *SIAM Journal on Optimization*, vol. 12, no. 4, pp. 979–1006, 2002.

[97] M. Diehl, D. B. Leineweber, and A. A. Schäfer, *MUSCOD-II users' manual*. Universität Heidelberg. Interdisziplinäres Zentrum für Wissenschaftliches Rechnen (IWR), 2001.

[98] S. Boyd and L. Vandenberghe, *Convex Optimization*. Cambridge University Press, Mar. 2004.

[99] A. Wächter and L. T. Biegler, "Line search filter methods for nonlinear programming: Motivation and global convergence," *SIAM Journal on Optimization*, vol. 16, no. 1, pp. 1–31, 2005.

[100] A. V. Rao, "A survey of numerical methods for optimal control," in *AAS/AIAA Astrodynamics Specialist Conference*, ser. Advances in the astronautical sciences, vol. 135. Univelt, Inc., 2009, pp. 497–528.

[101] B. A. Conway, "A survey of methods available for the numerical optimization of continuous dynamic systems," *Journal of Optimization Theory and Applications*, vol. 152, no. 2, pp. 271–306, Feb 2012.

[102] I. M. Ross and M. Karpenko, "A review of pseudospectral optimal control: From theory to flight," *Annual Reviews in Control*, vol. 36, no. 2, pp. 182 – 197, 2012. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S1367578812000375

[103] E. Trélat, "Optimal control and applications to aerospace: Some results and challenges," *Journal of Optimization Theory and Applications*, vol. 154, no. 3, pp. 713–758, Sep 2012.

[104] N. Natarajan, I. S. Dhillon, P. K. Ravikumar, and A. Tewari, "Learning with noisy labels," in *Advances in neural information processing systems (NIPS)*, 2013, pp. 1196–1204.

[105] S. M. Studios, *Project CARS*. Bandai Namco Entertainment, 2015.

[106] J. Davis, R. Johnson, and J. Stepanek, *Fundamentals of Aerospace Medicine*. Lippincott Williams & Wilkins, 2008.

[107] J. Frasch, A. Gray, M. Zanon, H. Ferreau, S. Sager, F. Borrelli, and M. Diehl, "An auto-generated nonlinear MPC algorithm for real-time obstacle avoidance of ground vehicles," in *Control Conference (ECC), 2013 European*, July 2013, pp. 4136–4141.

[108] R. Verschueren, M. Zanon, R. Quirynen, and M. Diehl, "Time-optimal race car driving using an online exact Hessian based nonlinear MPC algorithm," in *Proceedings of the European Control Conference (ECC)*, 2016.

[109] M. Gerdts, R. Henrion, D. Hömberg, and C. Landry, "Path planning and collision avoidance for robots," *Numerical Algebra, Control and Optimization*, vol. 2, pp. 437–463, 2012.

[110] M. Gerdts and I. Xausa, *Avoidance Trajectories Using Reachable Sets and Parametric Sensitivity Analysis*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 491–500.

[111] K. Bollino, L. R. Lewis, P. Sekhavat, and I. M. Ross, "Pseudospectral optimal control: A clear road for autonomous intelligent path planning," in *AIAA Infotech@Aerospace 2007 Conference and Exhibit*, 2007.

[112] M. Hurni, P. Sekhavat, M. Karpenko, and I. Ross, "A pseudospectral optimal motion planner for autonomous unmanned vehicles," in *American Control Conference (ACC), 2010*, June 2010, pp. 1591–1598.

[113] O. Khatib, *Real-Time Obstacle Avoidance for Manipulators and Mobile Robots*. New York, NY: Springer New York, 1990, pp. 396–404.

[114] M. Brännström, E. Coelingh, and J. Sjöberg, "Decision-making on when to brake and when to steer to avoid a collision," *International Journal of Vehicle Safety*, vol. 7, no. 1, pp. 87–106, Jan. 2014.

[115] J. Butcher, "A history of Runge-Kutta methods," *Applied Numerical Mathematics*, vol. 20, no. 3, pp. 247 – 260, 1996. [Online]. Available: http://www.sciencedirect.com/science/article/pii/0168927495001085

[116] A. Wächter, L. T. Biegler *et al.* Introduction to IPOPT: A tutorial for downloading, installing, and using IPOPT. Online. [Online]. Available: https://www.coin-or.org/Ipopt/documentation/node40.html

[117] B. Shahriari, K. Swersky, Z. Wang, R. P. Adams, and N. de Freitas, "Taking the human out of the loop: A review of Bayesian optimization," *Proceedings of the IEEE*, vol. 104, no. 1, pp. 148–175, Jan 2016.

[118] L. Breiman, J. H. Friedmann, R. A. Olshen, and C. J. Stone, *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[119] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed., ser. Springer Series in Statistics. Springer, 2009.

[120] N. Meinshausen, "Quantile regression forests," *Journal of Machine Learning Research*, vol. 7, pp. 983–999, Dec. 2006.

[121] L. Breiman, "Out-of-bag estimation," Statistics Department, University of California Berkeley, Tech. Rep., 1996. [Online]. Available: https://www.stat.berkeley.edu/~breiman/OOBestimation.pdf

[122] K. L. Campbell, "The SHRP 2 naturalistic driving study: Addressing driver performance and behavior in traffic safety," *TR News*, no. 282, pp. 30–35, 2012.

[123] A. Berthelot, A. Tamke, T. Dang, and G. Breuel, "Handling uncertainties in criticality assessment," in *Intelligent Vehicles Symposium (IV), 2011 IEEE*, June 2011, pp. 571–576.

[124] W. B. Powell, *Approximate Dynamic Programming: Solving the Curses of Dimensionality*. John Wiley & Sons, Inc., 2011.

[125] V. Vapnik, "An overview of statistical learning theory," *IEEE Transactions on Neural Networks*, vol. 10, no. 5, pp. 988–999, Sep 1999.

[126] T. O. Kvalseth, "Cautionary note about r2," *The American Statistician*, vol. 39, no. 4, pp. 279–285, 1985.

[127] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, pp. 1157–1182, 2003.

[128] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay, "Scikit-learn: Machine learning in Python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[129] B. Houska and B. Chachuat, "Branch-and-lift algorithm for deterministic global optimization in nonlinear optimal control," *Journal of Optimization Theory and Applications*, pp. 1–41, 2013.

[130] H. Borchani, G. Varando, C. Bielza, and P. Larrañaga, "A survey on multi-output regression," *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, vol. 5, no. 5, pp. 216–233, 2015.

[131] *Broad Agency Announcement - Explainable artificial intelligence (XAI)*, Defense Advanced Research Projects Agency (DARPA), Information Innovation Office, August 2016, accessed: July 29, 2017. [Online]. Available: https://www.darpa.mil/attachments/DARPA-BAA-16-53.pdf

[132] N.N., "Department of Defense World Geodetic System 1984: Its Definition and Relationships with Local Geodetic Systems," National Imagery and Mapping Agency, Tech. Rep. TR8350.2, 2000. [Online]. Available: http://earth-info.nga.mil/GandG/publications/tr8350.2/tr8350_2.html

[133] M. S. Grewal, L. R. Weill, and A. P. Andrews, *Global Positioning Systems, Inertial Navigation, and Integration*. John Wiley & Sons, Inc., 2007.

[134] B. Hofmann-Wellenhof, H. Lichtenegger, and E. Wasle, *GNSS — Global Navigation Satellite Systems: GPS, GLONASS, Galileo, and more*. Springer Vienna, 2008.

[135] R. H. Rapp, *Geometric Geodesy Part I*. Ohio State University Department of Geodetic Science and Surveying, April 1991. [Online]. Available: https://kb.osu.edu/dspace/handle/1811/24333