Simulation-based Methods for the Validation of Highly Automated Driving

Zur Erlangung des akademischen Grades eines

Doktors der Ingenieurwissenschaften

(Dr.-Ing.)

von der KIT-Fakultät für

Wirtschaftswissenschaften

des Karlsruher Instituts für Technologie (KIT)

genehmigte

DISSERTATION

von

Stefan Jesenski, M. Sc.

Tag der mündl. Prüfung:	28.08.2025
Hauptreferent:	Prof. Dr.-Ing. J. Marius Zöllner, KIT
Korreferent:	Prof. Dr.-Ing. Peter Vortisch, KIT

Waldenbuch, 25. Januar 2026

Abstract

In recent years, the automotive industry has striven to introduce automated driving functions (ADFs) of increasing complexity and capability. Traditional statistical validation methods like endurance runs, which were previously applied to prove automotive safety, reach their limits and become infeasible since society poses increasingly challenging safety demands to increasingly complex and capable ADFs. Simulations promise to deliver a valuable contribution to the resulting unsolved validation challenge, since they show beneficial properties like being reproducable, safe, controllable and fast. This thesis addresses some of the most pressing issues hindering the usage of simulations for validation purposes.

Firstly, a sufficiently well performing ADF fails seldomly. Hence, critical scenarios occur very rarely in statistical simulations and a high amount of simulation runs is needed to generate results with acceptable accuracy. Importance sampling (IS) was previously used to tackle this problem. IS skews the sampling of simulation runs to the areas of the parameter space which the ADF cannot handle, but allows determining unskewed statistical results. A typical approach to identify such critical areas in the context of IS is the cross-entropy method (CE). CE's optimization procedure scales badly with the complexity of modeled scenarios. This thesis introduces a new framework which allows identifying and limiting the CE optimization on relevant parts of parameter space. It is shown, that this enables scalable application of IS to complex scenarios as found on public roads.

Secondly, simulations require statistical models which represent the scenes in the surrounding of the tested ADF. Previously available scene models concentrated on simple road topologies like straight highways sections. Within this thesis, existing models are generalized to more complex topologies as intersections. Initially, the proposed model is based on Bayesian networks (BN). However, BNs scale badly with the number of learned parameters. Thus, the model is adapted to the usage of sum-product networks (SPNs). The proposed scene model is statistically evaluated and it is shown that SPNs increase the model's scalability.

Thirdly, it is shown how the proposed scene model can be combined with IS. The scene model's structure is adapted to incorporate parameters which address a scene's criticality. As it is proven, that enables CE to skew the model's inherent statistics in order to sample predominantly critical scenes and to quickly generate statistical evidence about an ADF's safety.

Kurzfassung

In den letzten Jahren war es das stete Bestreben der Automobilindustrie, immer komplexere Automatisierungsfunktionen (ADF) zu entwickeln. Bisherige Methoden der statistischen Validierung sind nicht mehr ausreichend, da gesellschaftliche Sicherheitsansprüche mit der Komplexität der ADF ansteigen. Simulative Methoden versprechen Abhilfe, da sie reproduzierbar, sicher, kontrollierbar und effizient sind. Diese Arbeit beschäftigt sich mit einigen der größten Hürden bei der Anwendung von Simulationen.

Der erste Teil behandelt die geringe Fehleranfälligkeit zulassungsfähiger ADF. Bei der Anwendung statistischer Simulationen führt dies zu nur selten auftretenden kritischen Szenarien. Somit wird eine große Anzahl an Simulationsdurchläufen benötigt, um statistische Aussagen mit hinreichender Genauigkeit zu berechnen. Importance Sampling (IS) wurde bereits zur Lösung derartiger Probleme verwendet. IS verschiebt die Statistik der Simulationsdurchläufe in kritische Bereiche des Parameterraums, kann aber unverschobene statistische Aussagen berechnen. Eine typische Methode, um solch kritische Bereiche im IS-Kontext aufzufinden, ist die Cross-Entropy Methode (CE). Der von CE benutzte Optimierungsansatz skaliert schlecht mit steigender Komplexität eines Szenarios. Diese Arbeit führt ein neues Framework ein, welches es ermöglicht relevante Parameterraumbereiche zu identifizieren und die CE Optimierung auf diese zu beschränken. Es wird gezeigt, dass dadurch die Anwendung von IS auf komplexe Szenarien, wie sie auf öffentlichen Straßen auftreten, möglich wird.

Der zweite Teil der Arbeit beschäftigt sich mit der statistischen Modellierung von Szenen in der Umgebung einer ADF. Bisherige Modelle beschäftigen sich mit einfachen Topologien wie geraden Autobahnabschnitten. Die vorhandenen Modelle werden hier auf die Modellierung komplexer Topologien wie Kreuzungen erweitert. Initial basiert das vorgeschlagene Modell auf Bayesschen Netzwerken. Aufgrund deren schlechten Skalierbarkeit mit steigender Parameterzahl werden sie durch Summen-Produkt Netwerke (SPNs) ersetzt. Das vorgeschlagene Modell wird ausgewertet und es wird gezeigt, dass SPNs die Skalierbarkeit signifikant erhöhen.

Im dritten Teil wird das vorgeschlagene Szenenmodell mit IS kombiniert. Hierfür wird es mit Parametern, welche die Kritikalität einer Szene beschreiben, ergänzt. Sie ermöglichen es, das Sampling von Szenen durch CE auf hauptsächlich kritische Szenen zu verschieben, was die schnelle Berechnung statistischer Aussagen zur Sicherheit einer ADF ermöglicht.

Contents

Nomenclature

Common abbreviations

Abbreviation	Description
ACC	Adaptive cruise control
ADF	Automated driving function
BM	Behavior model
BN	Bayesian network
CE	Cross entropy method
ConvLSTM	Convolutional long short term memory network
DAG	Directed acyclic graph
DCE	Distance-of-closest-encounter
DDT	Dynamical driving task
Destatis	German Federal Statistical Office (statist. Bundesamt)
HiL	Hardware in the loop
IB	Implemented behavior
IDM	Intelligent driver model
invTTC	Inverse TTC
IS	Importance sampling
KBL	Kullback-Leibler divergence
LSPN	LearnSPN is an algorithmic scheme to learn SPNs
MAE	Mean-absolute error
MCMC	Markov chain Monte Carlo
MCS	Monte Carlo simulation
MISE	Mean-integrated-squared error
MOBIL	Minimizing overall braking induced by lane changes
MSPN	MSPN is an augmentation of LearnSPN.
ODD	Operational design domain
OEDR	Object and event detection and response
OSSPN	OnlineSearchSPN is an online version of SearchSPN.
PET	Post-encroachment-time

Abbreviation	Description
PGM	Probabilistic graphical model
PRA	Proper response action
RB	Required behavior
RDC	Randomized dependency coefficient
RMSE	Root-mean-square error
RSS	Responsibility-sensitive safety
SB	Specified behavior
SSPN	SearchSPN is an algorithmic scheme to learn SPNs.
SiL	Software in the loop
SPN	Sum-product networks
STL	Signal temporal logic
SUMO	Simulation of Urban MObility
TSP	Time since intersection point
TTBo	Time to boundary
TTB	Time-to-brake
TTCE	Time-to-closest-encounter
TTC	Time-to-collision
TTP	Time to intersection point
TVD	Total variation distance
VeHiL	Vehicle hardware in the Loop
VRU	Vulnerable road user
WTTC	Worst-time-to-collision
XiL	X in the loop

Letters

Latin Letters

Symbol	Description
a	Integer describing the light phase of an intersection.
$\mathcal{B}$	Euler beta function
b	The rear end of the (virtual or real) predecessor vehicle when sampling between virtual vehicles.
b_0	The rear end of the last vehicle in the predecessor lane section.

Symbol	Description
$\mathfrak{C}$	Set of causality groups used to focus importance sampling on relevant simulation parameters.
C	Confidence level
d_b	Distance between the front boundary of the current lane (section) and the first vehicle
d_i	Distance between $(i-1)$th and ith vehicle in a lane (section).
d_pre	Distance to predecessor when sampling in the gaps between virtual vehicles.
$d_\mathrm{rel}^{(i)}$	Relevant distance to virtual vehicle i.
$d_{\mathrm{right},i}$	Distance between the predecessor in the lane on the right of the ith vehicle of a lane and its successor.
$d_{\mathrm{right},i}^{(\mathrm{ego})}$	Distance of a lane's ith vehicle to the predecessor in the lane on its right.
d_*	Distance to the next sampled vehicle when sampling between virtual vehicles.
$d_\mathrm{tot}^{(i)}$	Distance to virtual vehicle i.
d_TSP	The distance of the last vehicle after an intersection to the same intersection point.
d_TTBo	The distance of the first vehicle in a combining lane to the combining lane's front boundary.
d_TTP	The distance of the first vehicle before an intersection to the intersection point.
$\mathcal{F}_{\boldsymbol{f}_\mathcal{L}}$	Transformation of global coordinates in lane-related Frenet coordinates spanned by the centerline $\boldsymbol{f}_\mathcal{L}$ of lane $\mathcal{L}$
$\boldsymbol{f}_\mathcal{L}$	A function describing the 2-dimensional trajectory of the centerline of lane $\mathcal{L}$.
$\mathcal{F}_\mathrm{unit}$	Transformation into the unit range. Applied for Sensitivity Analysis.
$g_\mathrm{crit.}$	Microscopic safety metric usable to assess a simulation.
g_pair	Element of microscopic safety metric which is applied onto a relevant vehicle pair.
h	All simulation parameters in total. Includes properties with and without physical meaning.
$\mathfrak{i}$	Incidents
$\mathfrak{I}(\xi;\gamma)$	Indicator function which defines an event set.

Symbol	Description
$\mathcal{K}$	Relation between the original probability distribution and the skewed one when executing importance sampling.
k_{add}	Number of additional iterations for the cross entropy algorithm which are performed after the desired safety level is reached.
$k_{\max}$	Maximal number of iterations for the cross entropy algorithm.
$\mathfrak{Li}_{\log}$	Log likelihood
l_i	Length of the ith vehicle in a lane (section).
$\mathfrak{M}$	Computational model used for sensitivity analysis.
$\boldsymbol{m}$	All information which are required to describe a map and the traffic participants which populate it.
n	Distance in km
$\mathfrak{N}$	Number of simulation runs per iteration in algorithm 1.
N	Number of simulation runs.
$\mathcal{N}_0$	Seed SPN for Search SPN and Online Search SPN
$\mathfrak{N}_{\text{add}}$	Number of simulation runs which are performed for each of the k_{add} additional CE iterations which are executed after the desired safety level is reached.
N_{add}	Number of extra distributions added during the iterative cross entropy procedure to enable a more stable optimization.
$n_{\text{veh}}^{(\varphi_\xi)}$	The number of vehicles involved in simulation run ξ.
n_{φ_ξ}	Number of time steps for simulation run ξ
$q\left(\bullet\middle\vert\varphi_{\boldsymbol{\upsilon}}\right)$	Family of functions used to approximate the optimal importance sampling distribution.
r	Reliability
R	Reliability estimation
$\mathcal{R}\left(i,j,\xi,t\right)$	Checks a relation between vehicles i and j for simulation run ξ at time step t. It returns 1 when the relation is valid, otherwise 0.
s	Arc length position in the Frenet frame of a lane.
$\boldsymbol{\mathcal{S}}$	The set of ego vehicles which are relevant for the evaluation of a simulation run.
$\boldsymbol{\mathcal{T}}_{\varphi_\xi}$	Set of all time steps $\{0, \Delta t, ..., n_{\varphi_\xi} \cdot \Delta t\}$ included in a simulation run ξ.

Symbol	Description
t_a	The time since the traffic lights in the intersection have changed their phase.
v_i	Velocity of the ith vehicle in a lane (section).
v_{pre}	Velocity of predecessor when sampling in the gaps between virtual vehicles.
v_*	Velocity of the next sampled vehicle when sampling between virtual vehicles.
v_{TSP}	The velocity of the last vehicle after the intersection point on an intersecting lane.
v_{TTBo}	The velocity of the first vehicle in a combining lane.
v_{TTP}	The velocity of the first vehicle before the intersection point on an intersecting lane.
w_i	Width of the ith vehicle in a lane (section).
$\boldsymbol{x}_t$	Set of parameters defining a scene at time step t
z_0	Scaling factor for filtering when addressing the numerical instability of the weightings for importance sampling
z_1	Share of weightings around the median which are considered to be at least relevant when filtering to address the numerical instability of the weightings for importance sampling
z_2	Factor defining the relative distance to the next smaller neighbor which must be exceeded at the position of the filter cut when addressing the numerical instability of the weightings for importance sampling

Greek Letters

Symbol	Description
α	Threshold of significance for dependency measure.
β_i	Weighting of ith extra distribution which is added during the iterative cross entropy procedure to enable a more stable optimization.
$\boldsymbol{\chi}_t(i)$	Set of parameters defining vehicle i at simulation step t
$\delta_{i,j}$	Kronecker delta
Δ	Smoothing parameter for Laplace smoothing.
δ_{TVD}	Total variation distance

Symbol	Description
Δt	Time between two steps of a simulation
$\boldsymbol{\varepsilon}$	Set describing an event
$\Gamma(\bullet)$	Gamma distribution
γ	Boundary value for usage in an indicator function when defining an event set.
ι	The most recent samples used for training of Online Search SPN.
$\kappa_{\min}$	Minimal number of slices for instance split.
Λ	Values per batch for online learning algorithms.
μ	Original measure of sensitivity for the elementary effects method as published by Morris [73].
$\mu_{\text{campo.}}$	Modified measure of sensitivity for the elementary effects method as published by Campolongo et al. [19].
$\mu_{\mathcal{L}}$	Boolean variable deciding if a lane section $\mathcal{L}$ in an intersection is populated.
$\boldsymbol{\Omega}$	All possible scenarios which might be simulated
$\boldsymbol{\phi}$	Simulation parameters which do not have a physical meaning.
$\boldsymbol{\Phi}$	The space of all simulation parameters which do not have a physical meaning.
$\varphi_{\boldsymbol{X}}$	Full assignment to the random variables in $\boldsymbol{X}$.
Ψ_{global}	Global/intersectional parameters used to populate multiple lane sections when populating an intersection by an Incremental Roadway Population model.
Ψ_{local}	Local/intrasectional parameters used to populate single lane sections when populating an intersection by an Incremental Roadway Population model.
ρ	Lateral distance to the center line $\boldsymbol{f}_{\mathcal{L}}$ of a lane (Frenet frame).
σ	Type of leaf distribution.
$\boldsymbol{\nu}_{\varphi_\xi}$	The set of vehicles participating in simulation run ξ
ξ_i	Set parameterizing the physical parts of a simulation run. It consists of a consecutive entry of scenes
ζ	Relative distance related to the total distance to the successor when sampling in the gaps between virtual vehicles.

Superscripts

Index	Description
$(\bullet)^{\dagger}$	Superscript declaring a transposed matrix.

Mathematical operators

Operator	Description
$\times$	Cartesian product of two sets
$\mathbb{E}_{p(x)}[\bullet]$	Expectation value over distribution $p(x)$
$\perp\!\!\!\perp$	Independency
$\|\bullet\|_2$	Euclidean norm of a vector
$\oplus$	Element-wise sum of two sets
$\mathcal{P}(\bullet)$	Power set
$pre[i,t]$	Get the predecessor vehicle of vehicle i at time t
$Proj_{\mathcal{L}}[\bullet]$	Project a point onto the central line $\boldsymbol{f}_{\mathcal{L}}$ of lane $\mathcal{L}$.
$\mathfrak{q}_{\beta}(\bullet)$	Value at the β-quantil of a set of values.
$\sigma_{\text{dev,rel}}[\bullet]$	Relative standard deviation
$suc[i,t]$	Get the successor vehicle of vehicle i at time t
$\text{supp}(\bullet)$	Support of a function
$Val(\bullet)$	Get the space/assignments which are allowed for the random variables in the set on which Val is applied.
$\text{Var}_{p(x)}[\bullet]$	Variance over distribution $p(x)$

1 Introduction

This chapter gives an introduction to the thesis at hand. Section 1.1 provides a short summary of the motivation of the work. Section 1.2 gives an overview of the developed methodological framework, section 1.3 shortly lists the work's contributions to the state of the art and section 1.4 describes the structure and contents of the remaining chapters.

1.1 Motivation

In recent years strong technical and scientific progress towards the introduction of automated driving functions (ADF) has been made. However, the validation of these systems remains one of the key challenges since they must satisfy strong safety requirements. The existence of these strong requirements severely increases the efforts needed to validate an ADF by traditional methods like statistical testing on real roads by performing endurance runs. These traditional statistical approaches even seem to become infeasible (for more details see section 2.3) since an ADF must be able to handle all scenarios which it can possibly encounter. However for a safe system, the relevant critical scenarios are met very rarely when operating in the real world and large distances must be traveled in order to encounter enough of them for a reliable validation of the system. Hence, it is necessary to develop new holistic validation frameworks which again render the validation aims feasible. Since simulations possess positive properties like being reproducable, safe, controllable and fast, and since they include an omniscient observer for ground truth data, it is desirable and necessary to incorporate them into such new validation approaches.

However, there are strong hurdles to the application of simulations itself. One of the most prominent ones is the modeling of the highly complex surrounding of an ADF. The main reason for this challenge is, that the ADF must be able to operate in an environment which cannot be controlled by the ADF's manufacturer, which alters permanently and for which the total amount of possible events can never be fully specified. Hence, it

seems very difficult, especially when using simple parametric approaches, to statistically model this complexity. In the remainder, it is referred to the *open context* challenge of the ADF. This work tackles some of this problem's aspects by developing a machine-learning based method which enables the statistical modeling of time snapshots of the vehicle configuration in the surrounding of an automated vehicle. This method thereby can be used to create initial scenes for simulations of an ADF.

Another prominent challenge is inherent to the simulation-based creation of statistical evidence (see Definition 3.2) of ADF safety. As already mentioned, when testing an ADF on real roads, relevant critical scenarios will be encountered very rarely. This problem translates to the simulation-based creation of statistical evidence: For instance, when applying the very popular Monte-Carlo simulation (MCS) approach, one usually samples from probability density distributions which try to describe reality. Hence, a lot of the sampled scenarios will, as in reality, not be critical for the ADF and therefore the simulation's information density regarding the ADF's handling of critical scenarios will be very low. As a result, a very large number of simulation runs must be performed for stable statistical results. A popular procedure to resolve this problem is the application of Importance Sampling (IS) which emphasizes the sampling from critical parameter regions. IS maximizes the information density and the necessary simulation costs are therefore reduced. In literature, IS usually is affected by several limitations, especially by its bad scaling to the size of a simulation's parameter space. This work presents an IS approach which allows considering only the *relevant* parts of a simulation run and which thereby enables the application of IS to large and complex parameter spaces. The compatibility of the developed IS approach to the presented machine-learning based method to model time snapshots of the ADF's environment is then shown.

1.2 Concept overview

As already mentioned in the previous section, this work develops methods for efficient application of simulations to the challenge of generating statistical evidence for automated driving functions. The work is based on the concept of MCS. The blue parts of fig. 1.1 show the basic structure needed to execute MCS.

A normal MCS starts with a *stochastical scenario description model*. This model firstly parameterizes all elements which are necessary to fully

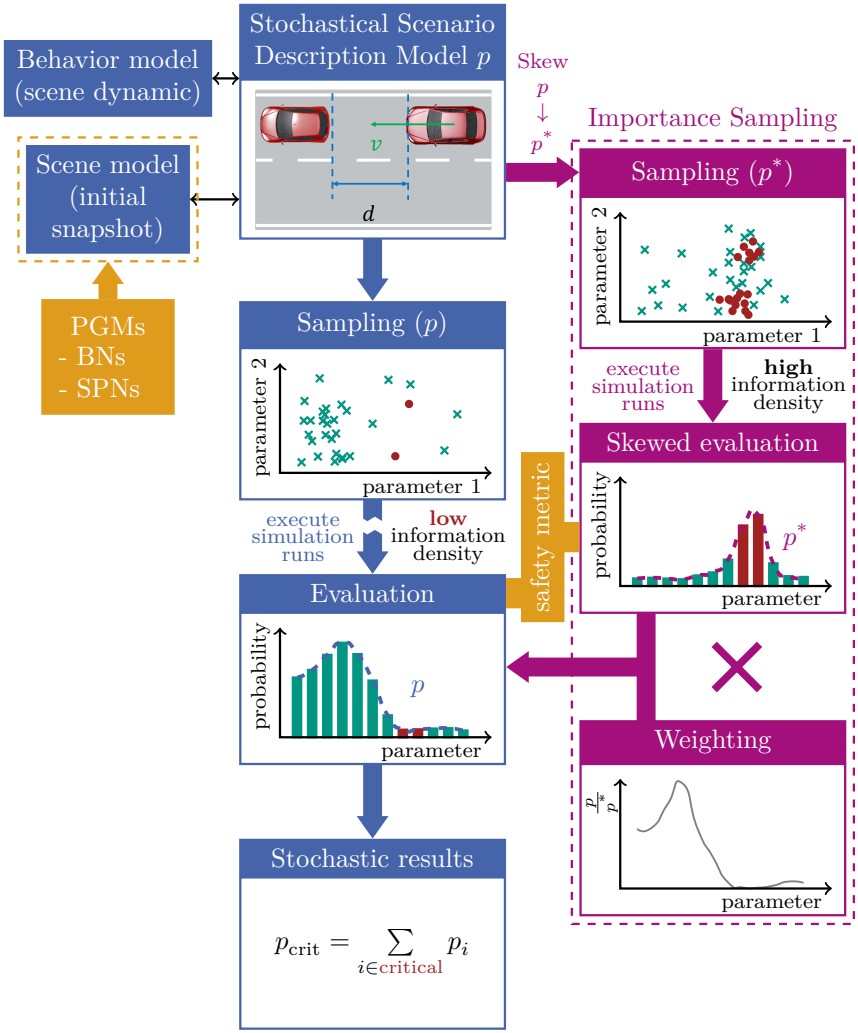


Figure 1.1 Modular structure of Monte Carlo simulations and illustration of application of importance sampling. The basic parts of MCS are marked in blue, whereas the parts related to IS are displayed in purple. The red circles in the sampling modules of MCS and IS mark critical scenarios which challenge the automated driving function under test. The same holds for the red bars in the "Evaluation" and "Skewed evaluation" modules. p describes the real probability distribution, p^* is the skewed importance sampling distribution which emphasizes critical scenarios.

specify a simulation run and secondly contains all necessary statistical information about the occurrence of the resulting parameter sets. These statistical information are usually encoded in (multivariate) parameter distributions p over the parameters. As far as this work is concerned, the stochastical scenario description model is separated into two sub-models: the *scene model* and the *behavior model*. The scene model statistically represents the initial time snapshot from which a simulation run starts, whereas the *behavior model* includes all the statistical information about the development and behavioral actions of the dynamic participants of a simulation run. As already mentioned earlier, this thesis discusses methods to represent the scene model. Since these methods must be able to represent an open context, the usage of non-parametric models which are not limited to pre-defined functional shapes is beneficial. This work therefore uses probabilistic graphical models (PGMs) since they allow modeling non-parametric multivariate probability distributions in a memory-efficient way and allow performing inference. That means they allow the derivation of conditional probability distributions in the form $p(a|b)$. As shown later in chapter 6, this is a basic requirement of our approach. Two distinct types of PGMs are discussed throughout this work: *Bayesian networks* (BNs) and *sum-product networks* (SPNs).

In the *Sampling* step of the MCS pipeline, parameter sets are sampled by using p. These parameter sets are then utilized to execute simulation runs which are used during the *evaluation* step to assess the criticality met during each simulation run. For that, metrics which quantify the level of criticality/safety are needed. Eventually in the *stochastic results* step, all simulation runs together are employed to extract statistical statements about the ADF under test.

For rare event simulations like the validation of a safe ADF, the MCS approach is well-known to be inefficient [12]. When having a look at the described pipeline the cause of this problem can be located in the sampling module. Since it is sampled from p which represents reality, only very few critical parameter sets are created as illustrated by the red circles in the sampling module in fig. 1.1. Thus, the information density concerning the system's behavior in critical scenarios is very low and in the evaluation module only little knowledge about safety critical behavior of the ADF can be gained (shown by the low red bars). As a result, the stochastic-results-module's statistical output will have a large variance unless a very large amount of samples from p are simulated. However, this would lead to high

costs and is inefficient (for details, see section 5.1.1).

A popular solution to this problem is importance sampling (IS) as described by the purple part of fig. 1.1. For IS, the *Sampling* module does not sample from p, but from a skewed distribution p^* which emphasizes critical parameter ranges. As a result, a lot of parameter sets which correspond to critical simulation runs are created. Hence, one will obtain a high information density and during the *skewed evaluation* step it is possible to obtain a lot of knowledge about the ADF's behavior in critical scenarios. At this point, the evaluation will of course be skewed and will not represent the stochastics of the real world represented by p. IS allows solving this problem by defining a weighting $\frac{p}{p^*}$ of the simulation runs. The application of the weighting transforms the skewed results back to results of the unskewed distribution p. While this allows the calculation of an unskewed result, the variance of the stochastical estimations will still remain small since more critical simulation runs have been observed. More information to the mathematical foundations and to the application of IS is given in section 5.1.2.

1.3 Contributions

The initial aim of this thesis was to contribute to the research regarding simulation-based methods for the validation of ADF. Therefore, it was essential to firstly gain insights how to formally describe the validation procedure, so that it can be evaluated which parts of the same can be supported by simulations. The 3-circles model published by Stellet et al. [138] gives such a formal description and categorizes validation into multiple sub-tasks.

Contribution 1 (Applicability of simulations to validation sub-tasks)
In order to formally understand how to apply simulations on the validation sub-tasks derived from the 3-circles model (see section 3.1.3), a translation between the terms of the 3-circles model and the technical terms of a simulation is derived in section 3.1.2.

To proceed further and to detect open challenges in the current state of the art, information from the formal understanding of validation were then

used to categorize and modularize the technical process of simulations for validation purposes. For each of the resulting modules an overview of the relevant literature was extracted.

Contribution 2 (Taxonomy of simulation pipeline)
A formal, modular description of the technical process of applying simulations for validation is given and an overview of the state of the art is discussed for each of the description's modules. See section 3.2.

The first concrete open challenge extracted from the state of the art is the modeling of scenes (time snapshots) in the surrounding of an ADF. The main challenge is the complexity of real traffic environments. The already existing models have a limited scope especially regarding the road topology. A more detailed discussion on the current limitations will later be given in section 3.3.1 (especially in section 3.3.1.5 for the chosen approach). This thesis is focused on logical parts of the ADF's surrounding. That means, no sensor noise or failures are included. However, it is of course possible to add imperfect sensor models in the methodological framework.

Contribution 3 (Extension of PGM scene model to intersections)
A PGM-based method for the construction of statistical scene models for the surrounding of an ADF [119] is extended to intersections and should in principle now be applicable to arbitrary road topologies. Section 6.2 covers this contribution.

In the beginning, the scene models were constructed by using BNs as proposed by Wheeler et al. [119]. This approach has significant limitations with regard to scalability to large and complex scenes since for Bayesian networks exact inference scales badly with an increasing number of parameters. A promising solution approach is the usage of sum-product networks since exact inference on the latter scales only linearly with the network size [32, Theorem 1].

Contribution 4 (Increase of efficiency of PGM-based scene model)
To the knowledge of the author, SPNs are firstly applied to the problem of scene generation. A comparison between SPNs and BNs concerning their efficiency and accuracy when representing a scene model is produced. For the comparison, several SPN learning algorithms are tested. It is shown that SPNs are much faster in producing scenes than BNs. This contribution is handled in section 6.3.

For the sake of efficiency, it is not sufficient to just sample from a statistical model which represents the real probability distribution p. In order to be able to sample from the IS distribution p^*, it is necessary to modify the PGM-based method to predominantly sample critical scenes.

Contribution 5 (Creation of scenes with a predefined criticality)
The PGM-based approach to the scene model is modified by adding a set of safety metrics to the learned probability distributions. Inference now allows sampling the scene's parameter set conditioned on the added safety metrics. By setting the safety metric parameters to a certain safety level, scenes with a certain criticality can then be sampled. Multiple safety metrics are compared. Special attention is paid to the sampling of critical scenes since they are most relevant for safety validation. More information in section 7.1.

As already described, IS is a popular method to enhance the efficiency of MCS for rare events. However, IS scales badly with an increasing size of the simulation's underlying parameter space. For instance the scenario illustrated in fig. 1.2 consists of an automated vehicle – which in the remainder is denoted as *ego vehicle* – and ten vehicles in its surrounding which are governed by a behavior model. Let the used behavior model be a simple parametric model, e.g. the intelligent driver model (IDM) [104] which includes seven parameters. When a skewed distribution p^* shall be approximated by a parametric family of functions consisting of truncated Gaussian distributions with three summands per behavior model parameter, an optimization to find a good approximation of p^* must be conducted in a space with $10\,\text{vehicles} \cdot 7\,\frac{\text{parameters}}{\text{vehicle}} \cdot 3\,\frac{\text{summands}}{\text{parameter}} \cdot 3\,\frac{\text{dimensions}}{\text{summand}} = 630\,\text{dimensions}$ which will not be feasible. By inclusion of other properties

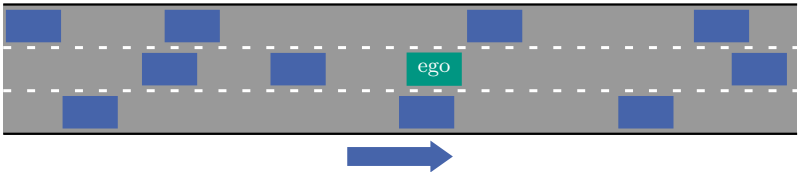


Figure 1.2 Scaling problem of IS. The number of parameters which must be optimized explodes with increasing complexity of a scenario. In this example ten vehicles are distributed in the near surrounding of the automated ego vehicle.

besides the behavior model, the number of dimensions will further increase. Contribution 6 circumvents this by introducing a methodology which allows limiting the IS optimization on selected parts of a simulation run.

Contribution 6 (Scalability of IS)
The problem of scalability of IS is tackled by the development of a method which allows limiting the IS optimization to find p^* on selected *relevant* parts of a scenario. As it is shown in sections 5.2 and 5.3, this method can strongly reduce the dimension of the optimization.

In order to apply this scalable method to IS optimization, it becomes necessary to assess which parts of a scenario are *relevant* with respect to the applied understanding of the ADF's safety.

Contribution 7 (Find relevant parts of a scenario)
The application of *Screening methods* from the area of sensitivity analysis should enable finding the relevant parts of a scenario. In section 5.4, it is shown that the *Elementary Effects method* is able to detect the relevant parameters of the stochastical scenario description model.

If the PGM-based scene generation method and the IS approach shall be combined, it must be shown that they are compatible and that IS can optimize the scene model towards the required critical parameter ranges.

Contribution 8 (Combination of PGM scene model and IS)
Section 7.2 shows that the developed scalable approach to IS is compatible to the developed PGM-based scene model in its advanced version resulting from contribution 5. A strategy to include the PGM-based scene model in the IS optimization is developed.

Quantitative metrics are essential for the definition of the terms "criticality" or "safety" which are assessed in the *evaluation* modules of the simulation pipeline. Until now, most metrics usually used in simulations are pure *physics-based* metrics which only assess physical criticality. However, *behavior-based* metrics which assess who is to blame for a safety violation are required (see also section 3.2.3). The model of *responsibility-sensitive safety* (RSS) [97] formalizes the allocation of blame. However, it does not directly give a metric whose derivation is therefore the next contribution.

Contribution 9 (Behavior-based metric)
A new behavior-based metric which considers blame, is derived from RSS in section 4.4.3. It is shown that the metric fulfills the requirements needed for application to IS. This is experimentally verified by applying the metric to the developed scalable IS framework in section 5.3.

A detailed comparison of the contributions 3 to 9 to the current state of the art is presented in section 3.3.

1.4 Outline

This work is structured into eight chapters. The chapters' placement in the overall context of the thesis is illustrated in fig. 1.3. Chapter 2 defines basic terms and concepts which are essential for the understanding of the work's aims and procedures. In detail, the terms "scenes" and "scenarios" (section 2.1) are explained. Additionally, an overview about the SAE categorization of automated driving functions is presented (section 2.2). Besides, the validation problem for highly automated driving functions and thereby this work's necessity is discussed (section 2.3).

In chapter 3, it is discussed how simulations can be used to support

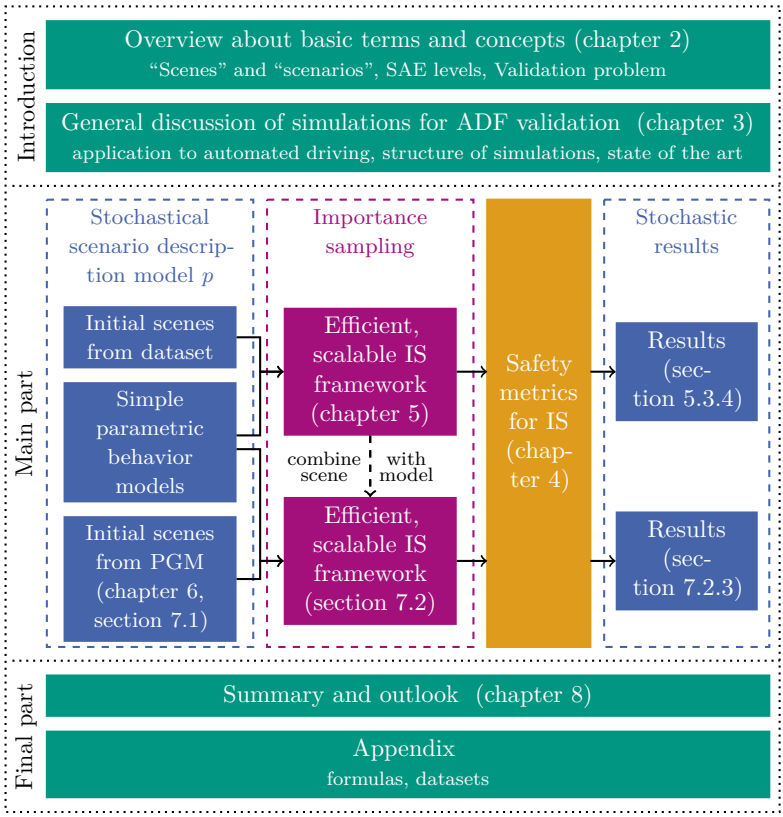


Figure 1.3 Rough placement of the chapters in the overall context.

solving this validation problem (section 3.1). In this relation, validation is formalized by using the 3-circles model. This allows identifying several validation sub-tasks which can be supported by simulations. Afterwards, the structure of a simulation is examined (section 3.2). That is, several required sub-modules of simulations are identified and described. Last but not least, the current state of the art with regard to this work's main concepts is analyzed which then directly illustrates the scientific relevance of the proposed concepts (section 3.3).

The evaluation module of simulations as identified in chapter 3 requires the definition of metrics to assess the safety of a simulation run. Therefore,

chapter 4 formally discusses safety metrics and their requirements (section 4.2). Four exemplary safety metrics are then introduced (sections 4.3 and 4.4). They will further be used in the remainder of the work.

Efficiency, which is one of the main challenges for executing rare-event simulations, is discussed in chapter 5. For that, a new scalable method for applying the cross-entropy method to find a good importance sampling distribution is introduced (sections 5.1 and 5.2). The method works by focusing on the relevant parts of a simulation run which have the greatest influence on an ADF's safety. The method is illustrated on a simulation framework which draws its initial scenes from a dataset and which uses simple parametric behavior models to describe the simulation runs' dynamics (section 5.3). Additionally, a method to identify the relevant parts of a scenario is given (section 5.4).

In chapter 5 it turns out that drawing the initial scene of a simulation run from a pure dataset implies severe disadvantages. Hence, in chapter 6, a scene model which allows generating these initial scenes is introduced. The discussed model, which is based on already existing work (section 6.1), is extended to be applicable to complex roadway topologies like intersections (section 6.2). Furthermore, a more efficient procedure to model the required probability distributions by sum-product networks is proposed (section 6.3).

Chapter 7 modifies the initial scene model from chapter 6 to make it possible to sample predominantly critical scenes (section 7.1). This allows its incorporation into chapter 5's IS framework. That mitigates certain disadvantages since the entire simulation run including the initial scene can now be optimized by the cross-entropy approach in order to find critical scenarios (section 7.2). This then enables getting more information about an ADF's safety in fewer simulation runs.

Chapter 8 summarizes the thesis and gives an outlook on still open questions and on possible future work.

Appendix A gives insights into the derivation of the formulas required in section 2.3 and appendix B gives an overview about the datasets which were utilized throughout this thesis.

2 Basics

In this chapter basic facts which are needed for further understanding of this work shall be discussed. In detail, definitions of the terms “scene” and “scenario” are given in section 2.1, a description of the common classification of automated driving functions into 6 levels is included in section 2.2 and a detailed derivation of the validation problem for highly automated driving systems is discussed in section 2.3.

2.1 Scenes & scenarios

Simulations for validation of automated driving functions typically consist of single simulation runs. It is necessary to describe accurately the contents of such a simulation run. In principle, a simulation run must include the environment which is *relevant* for the tested automated vehicle and starts with a *scene* and in total forms a *scenario*. In the following, the terms scene and scenario shall be defined as proposed by Ulbrich et al. [109].

Definition 2.1 (Scenes)

> “A scene describes a *snapshot* of the environment including the scenery and dynamic elements, as well as all actors’ and observers’ self-representations, and the relationships among those entities. Only a scene representation in a simulated world can be all-encompassing (objective scene, ground truth). In the real world it is incomplete, incorrent, uncertain, and from one or several observers’ points of view (subjective scene).” (Ulbrich et al. [109])

Following definition 2.1, a scene is a representation of the state of the relevant environment at one single time step (of a simulation). This representation includes information about all the static objects like trees, buildings,... (scenery), all the objects which have the ability to move like

footballs, buggies, ... as well as information about the states of the traffic participants (actors) and sensoric elements of the scene (observers). According to Ulbrich et al., the actors' and observers' self-representation even contain information which are not perceivable from the outside of the respective element. That can for example be information about skill levels or internal system states. As a result, it is obvious that an objective scene description can only be gathered in simulations where an "omniscient observer" (Ulbrich et al. [109]) is available.

Definition 2.2 (Scenarios)

"A scenario describes the temporal development between several scenes in a sequence of scenes. Every scenario starts with an initial scene. Actions & events as well as goals & values may be specified to characterize this temporal development in a scenario. Other than a scene, a scenario spans a certain amount of time." (Ulbrich et al. [109])

Ulbrich et al. [109] define a scenario to be a concatenation of scenes with certain actions and goals causing the transition between these scenes. In their definition a scenario does not necessarily have to be defined in every detail. In the extreme case they accept a scenario consisting of a detailed defined initial scene which then can develop based on set goals of the relevant actors.

However, for the purposes of this thesis, a scenario shall be viewed as a consecutive entry of fully defined scenes. Simulations usually consist of single, time-discrete simulation steps. The transition between these simulation steps will be governed by certain pre-defined behavior models which determine the dynamical behavior of the relevant actors. In this nomenclature, scenes are equivalent to a single simulation step and a scenario describes the fully defined, omnisciently viewed progress of a simulation run. Furthermore, the actions and events are handled by the actors' behavior models. An illustration of this connection is illustrated in fig. 2.1.

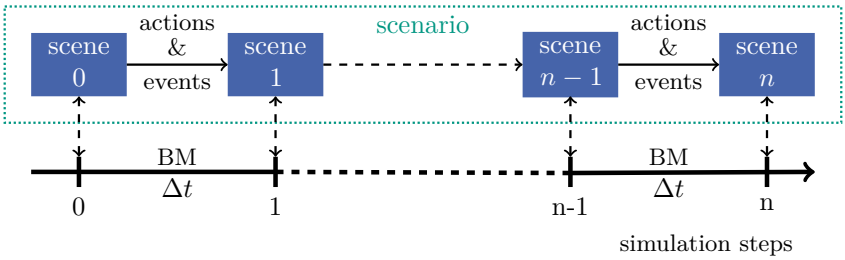


Figure 2.1 Illustration of correspondence of scenes/scenarios and the technical simulation procedure. The upper part shows the iteration of scenes and the lower part of the figure shows the corresponding simulation steps. BM is an abbreviation for behavior model, and Δt is the time length of one simulation step.

2.2 SAE levels

In 2014, SAE International (previously known as **S**ociety of **A**utomotive **E**ngineers) published a taxonomy on automated driving functions [47, last revision: 2021]. This taxonomy focuses on the subtasks of

- lateral/longitudinal *motion control*,
- object and event detection and response (OEDR) and
- providing a *fallback* to a malfunctioning ADF

which are necessary to ensure the correct execution of a dynamical driving task (DDT) [47, section 3.10]. Note, that in contrast to motion control and OEDR, the fallback function is not part of the DDT itself. Figure 2.2 gives a simple schematic overview of DDT's loop.

Motion control executes the longitudinal and lateral steering of the vehicle. It's outputs are based on interpreted and processed information about the vehicle's position and surrounding which are compared to the vehicle's operational goals. The outputs of motion control are quantitative steering commands which are sent to the vehicle's actuators.

The OEDR subtask encompasses the monitoring of the vehicle's environment by detecting, recognizing and classifying objects and events. In addition, it is responsible for reacting appropriately to these objects and events [47, section 3.19].

The *fallback* function acts if a *monitor* detects an issue in the ADF's behavior which endangers the fulfillment of the DDT. It must be able to

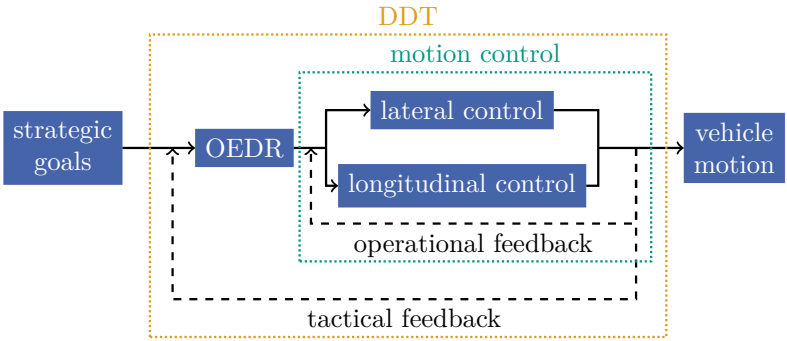


Figure 2.2 Schematic view of the DDT. The image is based on [47, figure 2]. For a definition of the terms *strategic*, *tactical* and *operational*, see [47, section 8.11].

Table 2.1 Overview over different levels of automated driving. This table is based on [47, table 1]. ODD is short for *operational design domain* which describes the area and driving modes the automated system can handle. Driver corresponds to a human being while system corresponds to a technical realization of a certain task.

level	Name	motion control	OEDR	fallback	ODD
0	No Driving Automation	driver	driver	driver	-
1	Driver Assistance	both	driver	driver	limited
2	Partial Driving Automation	system	driver	driver	limited
3	Conditional Driving Automation	system	system	driver	limited
4	High Driving Automation	system	system	system	limited
5	Full Driving Automation	system	system	system	unlimited

solve appearing safety critical situations. The primary focus of the fallback function is the safety of all traffic participants. It therefore aims to bring the system into a minimum risk condition. This could for example be implemented by doing a safe stop on a street's shoulder [47, section 3.12]. An illustration of these principles is given in fig. 2.3.

In the mentioned SAE taxonomy, automated driving functions are grouped into six levels of autonomy. An overview is given in table 2.1. For SAE-level 0 no automation takes place at all. That means the complete DDT is performed by a human driver. The driver obviously also has to supply the fallback and the complete OEDR. For level 1 the driving task is split. Either the longitudinal or the lateral motion control is executed by the system. For level 2, the system takes over both lateral and longitudinal control of the vehicle. However, for levels 1 and 2 at least parts of OEDR

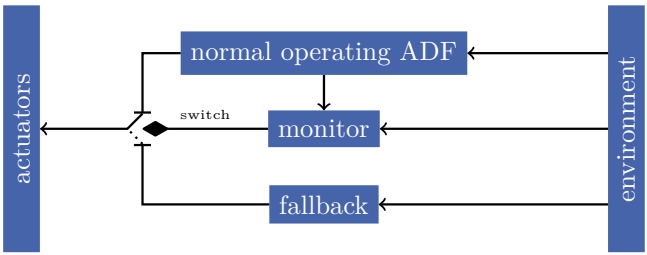


Figure 2.3 Simplified scheme of the functioning of a monitor. Let an ADF execute the DDT, possibly together with a driver. If the monitor detects some kind of failure/performance issue during the normal operation of the ADF, it activates the output of the fallback function, which is responsible to transition the vehicle into a minimum risk condition. ADF, monitor and fallback may get sensoric input from the environment. The ADF and fallback output control signals for the actuatoric layer of the vehicle.

and the fallback function must still be provided by the human driver. Hence, the driver must still monitor the correct functioning of the system.

This changes fundamentally at SAE level 3. Starting with level 3, the automated driving system itself must be able to execute the complete DDT by itself. Therefore, it must also be able to monitor itself and recognize (performance) issues and failures in advance. Having recognized a critical situation or operational design domain (ODD) limits, the system must inform the driver at an early stage, so that there is enough time for the driver to take over and to solve the situation. Consequently, the human driver still acts as fallback, but s/he doesn't have to actively monitor the execution of the DDT if not notified by the system. Hence, the driver is allowed having his focus on something different than driving if he is not explicitly alerted. Beginning at level 4, the system provides the fallback function of the driving task. No driver interference is needed anymore. For the levels 1-4, the ODD of the automated driving system is limited by a certain ADF-specific set of criteria. In contrast to that, level 5 allows the automated driving system to perform all driving, monitoring and fallback tasks at all possible locations and for all possible driving modes.

In recent years a lot of progress has been made regarding the implementation of automated driving functions in production vehicles. Currently, level 2 systems are in the market. The first level 3+ systems are following. However, as already mentioned, the implementation of level 3+ systems corresponds to a large paradigm shift: The system must execute the com-

plete DDT by itself for the first time. This corresponds with the ability to take over the monitoring of its own performance from the human driver. Therefore, the responsibility for the safe operation of the vehicle partly passes from the driver to the system and thereby to the developer of the vehicle.

It is of great importance for the introduction of newly developed automated driving functions, that these functions are thoroughly validated and tested. Therefore, the mentioned paradigm shift between level 2 and 3 and the associated increase of responsibility introduces great challenges to the validation of systems of level 3 and higher. The next section will give an overview over these challenges.

2.3 Validation problem for SAE levels 3+

As already mentioned in the previous chapter, the paradigm change between SAE level 2 and 3 introduces new challenges to the validation procedure. For level 1 and 2 functions the responsibility for the safe operation of the system lies purely with the driver. Therefore, these types of functions can be seen as *comfort* functions. Naturally, societal demands regarding the safety of comfort functions are way lower than demands towards systems of higher levels of autonomy. This allows a significant reduction in the necessary testing effort and enables the release of these low level functions by the use of traditional validation methods. Such traditional validation methods for example comprise testing vehicles for a certain distance on real roads (endurance runs) or testing a system on fully controlled test tracks. Since it is sufficient to test comfort functions for their operation in the most common scenarios, the distance necessary for release is still feasible.

This changes fundamentally when a safety critical function of SAE levels 3 and higher shall be introduced. Here, it is essential to test and validate the system to be able to handle all scenarios it could run into with a certain degree of reliability. One of the main challenges here is based on the fact, that the traffic environment the function must be able to operate on, is an almost infinitely large open context. That means one cannot fully specify and describe all the scenarios an automated vehicle could come across. That makes it very time-consuming if not even infeasible to assess the reliability of a driving function's handling of all these scenarios. Up to now, the question for the required level of reliability is not solved. Perhaps

this question should not fully be answered by the manufacturers of the automated driving functions themselves since it is a societal question of how much risk we are willing to tolerate.

Several approaches for a quantitative estimation of the effort required to traditionally validate a level 3+ driving function by using endurance runs have been given [51, 112]. The approach by Kalra and Paddock [51], which shall now be discussed, assumes that a requirement societies might demand from automated vehicles is at least, that they are more reliable respectively safer than current vehicles driven by human drivers (this is known as GAMAB principle, see section 2.2.3 in [50]). At this point, it is necessary to be very thorough with the analysis and distinguish several accident classes. For example, the German Federal Statistical Office (Destatis) distinguishes between 6 different classes of accidents[1] [110], which are

1. accident with persons killed (fatalities),
2. accident with seriously injured persons (heavily injured),
3. accident with slightly injured persons (slightly injured),
4. severe accident involving material damage in the narrow sense (severe),
5. other accident involving material damage,
6. other accident involving material damage under the influence of intoxicating substances.

The validation of an automated vehicle's reliability should be done separately for the accident classes. Otherwise, if only the total reliability of all accident classes together was considered, this could lead to an overall increase of reliability but a decrease for highly critical accident classes, e.g. the number of accidents involving material damage could be strongly decreased, however the number of accidents with persons killed could be increased at the same time and the total statistics would not show this. Society probably would not accept such an exchange of occurrences (in this context, ISO 26262 defines risk as the "combination of the probability

[1] This classification is not perfectly suitable for the assessment of safety. Especially class number 6 is not necessarily relevant. However, the first four classes are sufficient to get an estimate of the effort needed for traditional validation methods as shown later in fig. 2.4.

of occurrence of harm and the severity of that harm", therefore risk and occurrence probability cannot be considered separately).

Kalra and Paddock [51] address this point by separately calculating the number of necessary miles automated vehicles must be driven to get statistically significant statements about their safety for the accident classes of fatalities, injuries and total crashes. They utilized numbers from the US Bureau of Transportation Statistics. In the following, their procedure is transferred to numbers of Destatis. Their rough principle is to apply "success run statistics based on the binomial distribution" [79].

To execute the success run statistics a per-km reliability rate r is defined. The probability distribution of the expected amount of incidents $\mathfrak{i}$ when driving n kilometers is given by the binomial distribution

$$p\left(\varphi_{\mathfrak{i}} \middle| \varphi_n, \varphi_r\right) = \binom{\varphi_n}{\varphi_{\mathfrak{i}}} \varphi_r^{\varphi_n - \varphi_{\mathfrak{i}}} \cdot \left(1 - \varphi_r\right)^{\varphi_{\mathfrak{i}}} . \tag{2.1}$$

Formally, $\mathfrak{i}, n$ and r are random variables. Throughout this work $\varphi_{\boldsymbol{X}} \in Val\left(\boldsymbol{X}\right)$ in general describes a numerical assignment to the respective random variables in $\boldsymbol{X}$. The confidence level of the estimation R of the reliability rate based on this n driven kilometers can now be estimated by (see appendix A)

$$C(R|\varphi_n, \varphi_{\mathfrak{i}}) = \int_R^1 p\left(\varphi_r \middle| \varphi_n, \varphi_{\mathfrak{i}}\right) \, \mathrm{d}\varphi_r = 1 - \sum_{j=0}^{\varphi_{\mathfrak{i}}} \binom{\varphi_n}{j} (1-R)^j R^{\varphi_n - j}. \tag{2.2}$$

Kalra and Paddock [51] assume that during testing $\varphi_{\mathfrak{i}} = 0$ incidents happen. The number of kilometers φ_n which must be driven to achieve confidence level C can then be calculated by

$$\varphi_n = \frac{\ln\left[1 - C(R|\varphi_n, 0)\right]}{\ln(R)}. \tag{2.3}$$

Equation (2.3) shall now be used to calculate the necessary number of kilometers for different accident classes and for different desired confidence levels C. Since it shall be shown, that the automated vehicles are at least as save as a human driver, statistical values for accident rates on real streets and for the different accident classes are required in order to determine R. The values which are used in the following are given in table 2.2. The results of the calculations which give the necessary test distances are displayed in

Table 2.2 Accident rates as calculated by the values published by Destatis [110]. For the total mileage 757×10^9 km, as given in [93], is used. For the calculation of the accident rates it was further assumed, that on average two vehicles are involved in an accident.

accident class	rel. amount of accidents (in total 2 685 661)	accident rate $1 - R$ in $\frac{1}{\text{km}}$
fatalities	0.1 %	7.10×10^{-9}
heavily injured	2.1 %	1.40×10^{-7}
slightly injured	9.0 %	6.39×10^{-7}
severe	2.6 %	1.84×10^{-7}

fig. 2.4. The necessary amount of driven kilometers grows strongly with increasing confidence level. For a confidence level of 1, it diverges. However, even for lower confidence levels, n is already very high. For the fatalities, n encompasses several hundred million of kilometers.

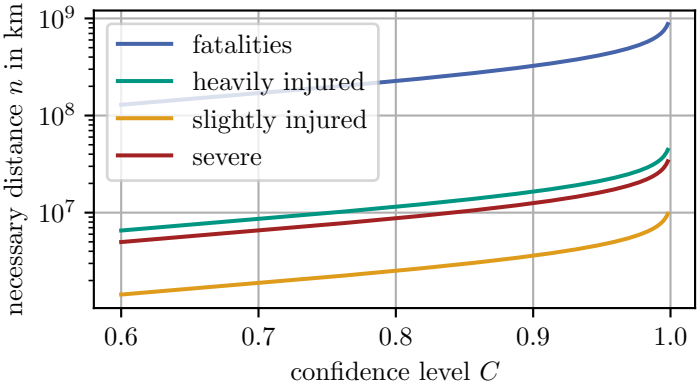


Figure 2.4 Number of necessary driven kilometers to reach a certain confidence level. For $C \to 1$, the necessary test distance diverges. The lower the accident rate, the higher n must be.

As already discussed by Kalra and Paddock [51], executing endurance runs over a distance of 10^8 km is not feasible. Not only might endurance runs of this length take several decades and be very expensive, it would also

be required to repeat them for each software version. For these reasons, it is absolutely essential to develop new holistic validation procedures which allow reducing these infeasible costs. Simulations have the potential to be part of such procedures since they offer several useful properties like reproducibility, controllability, efficiency, knowledge about the ground truth and safety during testing. This potential as well as still open questions concerning simulation-based procedures shall be discussed in the remainder of this work.

3 Simulations for Validation

In section 2.3 it was shown, that the validation of highly automated driving functions faces a feasibility challenge. In this chapter, it shall now be discussed how simulations could be used to form a part of the solution to this feasibility challenge. Section 3.1 recapitulates the *3-circles model* and uses it to derive statements about the areas of application of simulations, section 3.2 discusses the different models and parts which are required to run a simulation and section 3.3 debates the current state of the art and shows still open questions, which shall partly be addressed in the later chapters of this work. The work in sections 3.1 and 3.2 was already published by Jesenski et al. [133];

3.1 Possible areas of application

Validation is a complex task and therefore a thorough description of the validation process becomes necessary. In this work, the validation process shall be described by utilizing the *3-circles* model introduced in [82, 138] (see section 3.1.1). This abstract description of the validation process is used to give a short, abstract overview how to implement simulations within the validation process. This is accomplished by discussing translations between the terms of the 3-circles model and the technical terms of a simulation run (see section 3.1.2). Additionally, the formal description of the validation process allows characterizing several validation subtasks (see section 3.1.3).

3.1.1 3-circles model

In this chapter, the 3-circles model [138] is shortly recapitulated and extended to the application on simulation-based methods. In total, the complex task of building a *valid* automated driving system can be characterized by the *validation triangle* as given in fig. 3.1. An automated driving system is valid if the validation triangle is *consistent*, i.e. the system's *realization* must fulfill its *purpose* on its *context*. The context of a system is

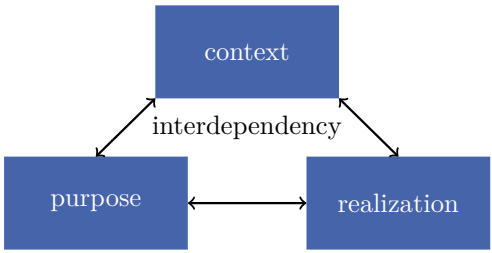


Figure 3.1 A validation triangle consists of three interdependent elements: Context, purpose and realization. The aim of validation is to show that the validation triangle is *consistent*. That means it is essential to show that a system's realization fulfills its purpose within the context it operates on. The three elements are highly interdependent on each other.

the domain, i.e. the relevant parts of the environment, the system must be able to operate on. As an example, that could be the relevant information about one lane streets in a city where a robotaxi shall be deployed. In the same example, the purpose of a system could be to provide a robotaxi service and the realization would be the technical implementation of the robotaxi itself. These elements are strongly interdependent. A system's realization is of course dependent on its purpose, e.g. a vehicle which shall transport a large number of people might look like a bus, whereas a vehicle which should only transport few people might be much smaller. The same is also valid for the realization's relation to the context. A vehicle which aims to transport people over a river must look different than a vehicle transporting them through a forest. Of course, the same also holds for the purpose of a system, which strongly depends onto context and realization. Even the context of a system is dependent on the other two elements. As an example, the relevant contextual information about the environment strongly change when using different sensors for the realization. When using a camera, colors matter a lot, however when using a Lidar sensor they won't even be recorded.

Another challenge included in the validation triangle is the complexity of its constituents. It will probably not be possible to explicitly describe either of them. The reasons for that are eclectic. Firstly, the context a real world system must be able to operate on will be very complex, since the world is unstructured and one just cannot completely describe everything which could happen on real roads. Additionally, the context will change

since infrastructure, traffic participants and other influences will change over the years of operation. The same also holds for the purpose of a system since implicit assumptions are often included in a system's definition. A reason for that is for example that basic working principles are given qualitatively and not quantitatively and these statements can often be strongly correlated to the situation a system is running into. Exemplarily, it is not easy to translate a purpose like "drive safe and fast to the destination your passenger wants to reach" into quantitative instructions. A similar argument for the system's realization is its *emergent* behavior [138]: Since a driving system will be very complex and consists of different subsystems, it will not be enough to analyze the subsystems since the system's behavior will be shaped by (unexpected) interactions between these subsystems.

As a result of these deficiencies, it becomes necessary to distinguish different types of *behaviors*. A behavior describes the interactions between the system and its environment [138] which is given by the entirety of a validation triangle. Stellet et al. [138] distinguish between three different types of behavior which are

- required behavior (RB),
- specified behavior (SB), and
- implemented behavior (IB).

RB describes the infinitely complex behavior which the developer tries to achieve. It is the behavior which is required in reality. SB gives the formalized part of the RB, which was explicitly understood and is written down in e.g. a specification which is used during the development process. IB characterizes the behavior which eventually was achieved after the implementation of the system. Because of the possible emergent behavior and the complexity of the system, the implemented behavior will deviate from the system's specification. Each of the behavior types contains a validation triangle by itself.

The elements of the validation triangle of RB shall be called ∞-*complex context*, *aimed purpose* and *required realization*. Since the aimed purpose strongly relies on implicit assumptions and the ∞-complex context must describe an infinitely complex ODD, they cannot be described in a formally complete way. It is important to note that a valid system is not allowed to leave its ODD. The elements of the validation triangle of SB are called "expected to be relevant" context, intended purpose and specified realization.

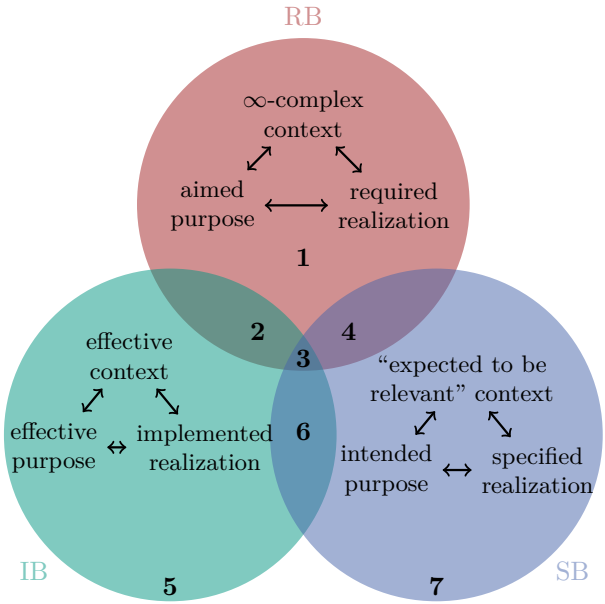


Figure 3.2 3-circles model describing the important entities necessary for validation of ADFs. The figure is based on a figure from [138] but was extended by additional information on the relations between validation triangles and behavior sets. Image published by Jesenski et al. [133], © 2019 IEEE.

Of course, these elements can be expressed explicit and complete since they are a specified and simplified approximation of the incomprehensible elements of the triangle of RB. Equivalently IB comprises the effective context, the effective purpose and the implemented realization.

The aim of the validation procedure must be to show that the overlap of these three behaviors, which is represented by cut-set 3 in fig. 3.2, is large enough. All the other cut-sets and areas represent failures. The areas 1, 5 and cut-sets 4, 6 probably will have the worst effects since here the system is not implemented as required. However, also cut-set 2 and area 7 can have bad effects since here the correct/false behavior was implemented/not implemented by chance and not as a result of good system knowledge. This can at least have bad implications in later system revisions. As will be shown in the next sections, it might be possible to get information about the size of cut-set 3 by using simulations to generate *evidence* about the consistency

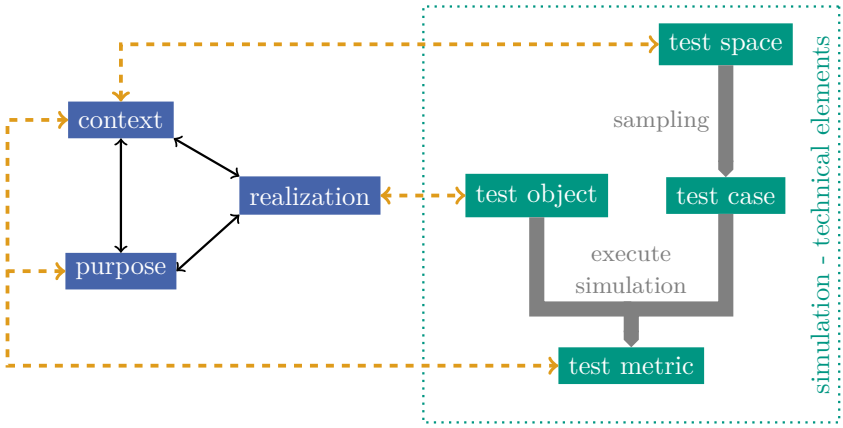


Figure 3.3 Relationship of the terms of the validation triangle to the technical terms of a simulation-based procedure. The simulation procedure describes the elements of one simulation run and is simplified. For a more elaborate discussion on simulations see section 3.2.

of certain validation triangles. In order to proceed, an understanding of the different types of evidence and how it can be generated by the use of simulations becomes necessary.

3.1.2 Types of evidence

Simulations are useful to generate evidence that a validation triangle is consistent. As already discussed, consistency implies that a certain realization can fulfill its purpose on the relevant context of the validation triangle. In order to apply simulations, it is first of all necessary to translate the terms used in the 3-circles model to the technical terms and modules available in simulations. The translations are given in fig. 3.3. Therefore, evidence is generated by sampling scenarios (test cases) from a test space (representation of context). These test cases are then used to evaluate a test object (representation of realization) by applying a test metric (checks the fulfillment of the purpose for the context). It is possible to distinguish two types of evidence: *System knowledge* and *statistical evidence*.

Definition 3.1 (System knowledge)
To gain system knowledge means to understand how an automated vehicle reacts and behaves in different scenarios. Therefore, in order to generate evidence of this kind, it is essential to simulate as many diverse test cases as possible since the main aim is to find the areas of the test space which are the most critical. System knowledge can be gained by the application of *microscopic metrics* (see section 3.2.3).

Definition 3.2 (Statistical evidence)
Statistical evidence allows deriving statistical statements about the appearance of errors in the validation triangle of an automated system. As an example, it could be the aim to derive a collision rate which then could be compared to the collision rates of human drivers. Statistical evidence can be generated by the application of *macroscopic metrics* (see section 3.2.3).

3.1.3 Areas of application

As already mentioned, simulations can be used to generate evidence about the consistency of a validation triangle. However, which validation triangles are relevant and can be used to give hints about the overlap of the behaviors as illustrated in fig. 3.2? To understand this, it is necessary to analyze which validation sub-tasks are implied by the 3-circles model. As a result of fig. 3.2, it is necessary to perform

1. Verification (SB $\implies$ IB),
2. Validation of the specification (RB $\implies$ SB), and
3. Validation of implementation (RB $\implies$ IB).

Each of these sub-tasks addresses the overlap of two of the three types of behavior.

Verification is a well-known problem in the development of complex systems. It is used to show that the implemented system fulfills the explicit specification used for development. Formally, such a simulation tests the consistency of the triangle of implemented realization, “expected to be relevant” context and intended purpose, as displayed in fig. 3.4. That means,

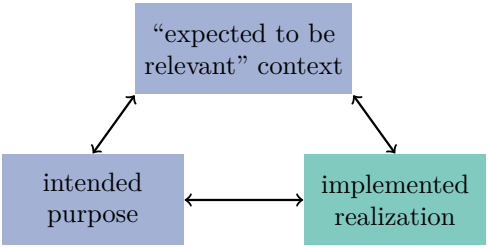


Figure 3.4 Validation triangle used by simulations in the verification sub-task.

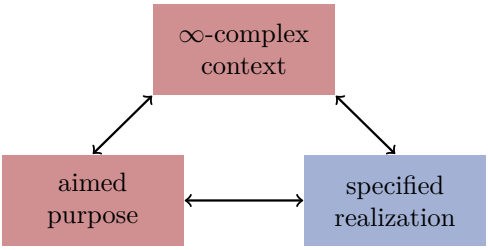


Figure 3.5 Validation triangle which is utilizable for the validation of the specification.

a validation triangle newly constructed from parts of the triangles of the behaviors is simulated and validated.

The validation of the specification must find evidence that the explicitly described specification of the system matches the stakeholders' implicit expectations. In other words, it must be shown that the system's specification is in accordance with the requirements of the ∞-complex reality and therefore specifies a truly “useful” and “safe” automation. Analogous to the verification task, the validation triangle shown in fig. 3.5 can be assigned to this subtask. The simulations must provide evidence that the validation triangle of specified realization, ∞-complex context and aimed purpose is consistent.

The validation of the implementation sub-task is equivalent to the classical validation approach. Here, it is assessed if an implemented system fulfills the real needs of the stakeholders, whereas these needs might not be expressed explicitly. In summary, the system implementation is compared to the demands of reality as given by RB. Figure 3.6 shows the associated validation triangle. The aim is to show the consistency of the triangle

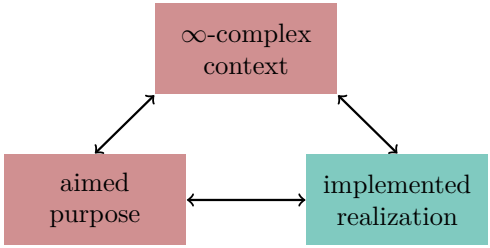


Figure 3.6 The Validation triangle applied to the validation of implementation.

of implemented realization, ∞-complex context and aimed purpose. In addition, it is possible to use simulations for further tasks like showing the *consistency of the specification* itself.

A more detailed assessment of the validation sub-tasks with their scopes, limitations and challenges goes beyond the scope of this work. The interested reader may have a look at the discussion published by Jesenski et al. [133] in 2019.

3.2 Structure of simulations & related work

Simulations which can be used to generate evidence for the consistency of the validation triangles of the validation sub-tasks exhibit a certain structure. In this section this structure and the relevant properties of the simulations are highlighted. The basic structure used in this work to differentiate the simulations is presented in fig. 3.7. In principle, it is possible to distinguish three phases in the entire simulation procedure. During the *test case generation* phase a *sampling method* is used to sample a parameter combination from the parameter ranges of the models which describe the *test space*. During the *simulation run*, the *test case* defined by the parameter combination is used to check the response of the *test object*. The response of the test object is then assessed in the *evaluation* phase. The evaluation of a single simulation run is achieved by the use of a *microscopic test metric*. If a stopping criterion is met, a total final *statement on all simulation runs* is determined by a *macroscopic test metric* and the simulation is stopped. Otherwise, the phases are repeated and another simulation run is conducted. Due to the emergent behavior of an automated vehicle, simulations considering the entire driving stack could be

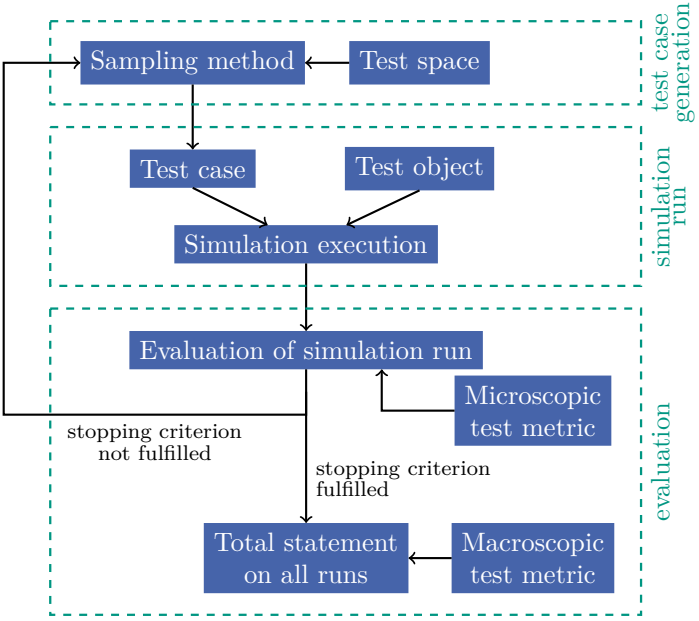


Figure 3.7 Simulation-based procedure for validation. A simulation run can be categorized into the three phases of *test case generation, simulation run* and *evaluation*. Simulation runs are conducted until a stopping criterion is fulfilled. This stopping criterion could be a statement about the completeness of the tested space or a statement about the accuracy of a statistical result. Graph based on Figure 4 published by Jesenski et al. [133], © 2019 IEEE.

highly beneficial. For this reason, the following discussion of the parts of the procedure focuses on the consideration of system-level simulations. Each of the following subsections debates one of the three simulation phases.

3.2.1 Test case generation

3.2.1.1 Sampling methods

A sampling strategy is needed to make sure that the results of the simulation have a sensible meaning and are calculated efficiently. For highly automated driving functions, which are usually open context systems, it probably will not be possible to sample the test space exhaustively. Hence, the sampling

strategy must be able to select relevant test cases/samples which allow determining high fidelity simulation results with as few samples as possible. In particular, the sampling methods can be divided into two categories: *Coverage-based sampling* and *statistical sampling*.

Coverage-based sampling is based on the principle to sample as many different parameter sets as possible. Consequently, this approach allows generating system knowledge (see definition 3.1). That means, one generates insights into the system and explores possible interactions with the system's environment. Lots of different designs for sampling strategies of this type are possible. In principle, one can sample

1. evenly distributed or in some other geometrical order over the whole test space, or
2. restricted to interesting critical parts of parameter space.

Regarding the first approach, an overview about suitable combinatorial methods and coverage criteria, especially for the field of software testing is presented by Grindal et al. [37]. Schuldt [94] utilizes combinatorial methods in combination with equivalence classes and boundary value analysis. Rocklage et al. [87] and Tuncali et al. [107] apply combinatorial t-wise sampling approaches which make sure that all combinations of t-sized parameter subsets are tested.

In relation to the second approach, that means when aiming to restrict the sampling to error-prone areas of parameter space,

- optimization-based [13, 106, 107],
- learning-based [48], and
- search-based [2, 4]

methods can be used to find relevant, critical parameter combinations. These search/optimization-based methods usually require a high number of computationally expensive simulation runs. Therefore, surrogate models can be used to approximate the criticality-related results of simulation runs [101, 106]. Hence, the number of directly executed simulation runs might be significantly reduced and the search for critical behavior can be fastened.

Recently, implementations which try to find error-prone parameter areas based on the usage of dynamic programming [22], on exploration via

rapidly-exploring random trees [60, 105] and on the application of adaptive stress testing [59] have been proposed. Especially adaptive stress testing has drawn a lot of attention [23, 49, 57–59]. Here, the problem of finding collision scenarios is solved based on the description of a simulation run as a Markov decision process. Optimization algorithms are then used to find a policy which optimizes a specific reward function on this decision process. This reward function is related to the probability of actions leading to a collision and thereby allows the optimization of the most probable collisions. In addition to these approaches, methods utilizing reachability analysis are also viable [7, 53]. Besides, methods to find feature interaction failures [3], methods to find the safety boundaries of error-prone regions of parameter space [11, 75, 130], methods to vary real, recorded scenes [131] and approaches to derive test cases from test models [43] exist. Furthermore, Corso and Kochenderfer [21] define an optimization-based approach which enables finding human-interpretable signal-temporal-logic-based failure descriptions. Abbas et al. [1] propose the combination of optimization-based test approaches and formal verification.

It is often hard to define a stopping criterion for such coverage-based sampling methods since a metric which defines when "enough" samples have been drawn is difficult to find [94]. Without the addition of some statistical representations, coverage-based methods usually do not allow generating statistical evidence (see definition 3.2). Another disadvantage is that these approaches often work by falsification which means that only single failures, often the most probable failure, and not the entire range of errors is investigated.

Statistical sampling methods are applicable when one desires to determine statistical evidence (definition 3.2) like accident rates of an automated system. Statistical sampling methods sample from probability distributions which must be defined by test space models (section 3.2.1.2). A strong challenge with statistical sampling methods is that they usually sample mostly in highly probable areas of the test space. However, these areas usually are understood quite good because they happen very often in the real world. That means, the amount of samples which are necessary to get new insights into the system and its critical areas, which mostly lie in very unlikely sections of the test space, is pretty high. Consequently, the argumentation described in section 2.3 partly also holds for simulations.

Different techniques which allow to mitigate this challenge have been

developed [12]. Examples are importance sampling [30, 33, 127] or subset sampling [78, 124]. The basic idea behind these algorithms is that they allow sampling from skewed probability distributions which especially emphasize critical parts of parameter space and include a procedure to calculate the generated statements back to the original, unskewed distribution. For IS this skewed distribution is denoted as *optimal importance sampling distribution*. The Cross-Entropy method [15, 89] is often used to optimize the optimal importance sampling distribution [80, 91, 126]. Detailed information on the mathematical formulation and the ideas behind IS and the cross-entropy method can be found in section 5.1. Wang et al. [116] use reachability analysis to define the criticality of parameter ranges of a crosswalk scenario. This allows creating an importance sampling distribution which emphasizes only "feasible test cases" which are caused at least partially by the automated ego vehicle. Wheeler and Kochenderfer [121] present a method to cluster critical scenes and model them by utilizing factor graphs. By sampling from these critical factor graphs, importance sampling becomes possible. Huang et al. [45] propose a method to use surrogate functions based on Kriging models to further enhance importance sampling. Uesato et al. [108] estimate the optimal importance sampling distribution by using failures happening during the training procedure of machine learning agents. By doing so they exploit the fact that during training the agents will still cause more failures than when they are properly trained. Using this to validate the fully trained agent however necessitates the assumption that the types of possible failures do not change during the training procedure.

In principle, these methods all pursue the same objective: the reduction of variance of the estimations of the statistical values when the simulation is limited to a certain budget of simulation runs.

A stopping criterion for statistical sampling methods can be based on the variance of the statistical results of the simulation [33]. On top of the discussed variance reduction methods, Huang et al. [46] propose a method to estimate the input uncertainty – which is caused by the variability of training data – of the statistical models used as inputs for Monte-Carlo simulations.

3.2.1.2 Test space

The test space describes the entirety of test cases/scenarios which the system under test might encounter. System-level simulations' test space mainly considers the surrounding of an automated vehicle. As already

mentioned, the surrounding of automated vehicles (∞-complex context, RB) usually implies an open context problem. Thereby, it is only possible to model an approximation of the context. The open context includes a large variety of static and dynamic entities in the vehicle's surrounding. To handle this variety, Schuldt et al. [95] introduce a model which allows structuring and categorizing these entities into 4 different hierarchical layers. The model later was extended to 6 layers [92].

The creation of models which describe the test space usually requires the

1. selection of features/parameters which shall be modeled [38],
2. decision about parameter ranges and distributions over these parameters which can be used for the sampling,
3. handling of the continuum of values, possibly the discretization of the parameter ranges. In the simplest case equidistant bins can be used [119]. Chen et al. [20] discuss a more elaborate scheme applicable especially to Bayesian networks.

Schuldt [94] also gives a high-level discussion of the generation of the test space and the inherent demands. The literature on test space modeling can be distinguished into the two categories of *generic modeling* and *maneuver-based modeling* (these terms will be discussed in the next paragraphs). Another important property which strongly influences the structure of a test space model is *the availability of statistical information* about the parameter values enclosed in the models and the structural organization (e.g. *distribution-based* vs *distribution-free*) of the same.

Since the validation of automated vehicles benefits by the use of closed-loop simulations (see section 3.2.2), test space models often decompose into a static scene model (e.g. [119]), which creates the initial scene of a simulation run and behavior models (e.g. [104]), which describe the dynamics of objects in the surrounding of the automated vehicle under test. Some works also attempt to use hybrid models which unify scene and behavior models. However, such hybrid approaches often are not as generic and flexible as the separated approach, e.g. the hybrid approach in [127] only models one particular scenario. Another possible method to tackle the infeasability-problem of the open context is the concept of the so-called "shadow mode" as described by Koenig et al. [55], Rocklage [86], and Wachenfeld and Winner [111]. Here an automated driving function is implemented, but is only running passively in the background to analyze

the scenarios the car experiences. Hence, it is possible to select relevant scenarios where the function would possibly fail. The open loop of the analysis of the scenarios can be closed by utilizing offline simulations.

Maneuver-based modeling describes the creation of models which only model one particular type of traffic maneuver. For instance, that can be models of car-following scenarios [30, 127], cut-in scenarios [126, 128, 129], lane departure events [115] or parking scenarios [18]. Such maneuver-based models can be advantageous since they are easier to design and need less parameters than generic models and they usually have an expressive meaning. However, a large disadvantage is that it is necessary to define an extra model for each maneuver type included in reality. That is, the ∞-complex context is not represented by one model c but by a union $c = \bigcup_i c_i$. Because of the open context, the number of necessary models is very high. Hence, it is essential to create some sort of scenario catalog which comprises all relevant scenarios [27, 30]. However, the creation of a complete scenario catalog may be time-consuming or may even be impossible. Basic approaches to scenario catalogs are documented in [103, 129]. Additionally, there are data-driven methods [66, 125] and formal approaches, eg. based on ontologies [10, 54], for catalog creation.

Generic modeling approaches enable the modeling of the test space without being limited to certain maneuver types. As a result, a scenario catalog is not necessary since it should be possible to obtain scenarios of different maneuver types by just sampling from the model's parameter space. Possible disadvantages of a generic modeling approach are parameter explosion, less expressive parameters and the difficulty to be accurate and complete on the ∞-complex context. Examples for generic modeling techniques are given by [64, 80, 118, 119]. A more detailed discussion on the state of the art regarding the generic modeling of scenes is included in section 3.3.1.

The availability of statistical information is essential when a test space model shall be used for statistical sampling as defined in section 3.2.1.1. The development of statistical models is a large challenge and requires the availability of large amounts of data targeting the operational design domain of the automated driving function under test.

In general, it is possible to distinguish statistical models relying on a particular parametric family of probability distributions and statistical models which work distribution-free.

Distribution-based statistical models usually benefit from few, but expressive parameters. Besides, the models' output usually is comprehensible. As a disadvantage, the structure of such models is fixed and thereby limited in its ability to represent the open context in the surrounding of the automated system. Distribution-based models often are split up into models for lane-following [36, 104, 122], lane change [28, 52] and gap acceptance. Bonsall et al. [16] discuss these models and some of their main parameters.

Distribution-free statistical models are not limited to a certain family of distributions. That means, they are more flexible in their structure. Since it might be hard to find a certain class of functions which sufficiently represents the open context of an automated function under test, the usage of distribution-free methods is indicated. A comparison of the accuracy of distribution-based and distribution-free models regarding speed prediction is given by Lefevre et al. [65]. Distribution-free models often contain a larger number of parameters which cannot be understood semantically, e.g. it is generally not possible to naively understand how a change in the weighting of a specific neuron of a large neuronal network influences the network's output.

For instance, distribution-free models can be created by Bayesian networks [35, 119, 120] (see section 6.1.2), sum-product networks (see section 6.1.3), factor graphs [118], tree diagrams [120] and neuronal networks [74].

When handling sequential decision problems like the behavior of traffic participants, specific distribution-free techniques like generative adversarial imitation learning should be used. In contrast, when naively using traditional supervised learning methods which are trained on (input, output) tuples for each single decision in a sequential decision process, small errors in each decision can sum up and hence the entire modeled decision sequence can produce catastrophically false results. As already mentioned, this problem of cascading errors can be addressed by specific methods like adversarial imitation learning and reinforcement learning [63, 64] since these methods do not train on single decisions, but on the entire sequences at once.

3.2.2 Simulation run

3.2.2.1 General properties

Dependent on the test object and the aims of the simulation several general properties of a simulation run can be adjusted.

Granularity: The Granularity is one of the most important properties of a simulation run. One distinguishes macroscopic, mesoscopic, microscopic and sub-microscopic simulation granularity [62, 67]. For the validation of automated systems usually a sub-microscopic granularity is desirable since it allows considering vehicle sub-structures like single hardware components. In contrast, macroscopic simulations only consider "thermodynamic-like" sizes as traffic flows. Microscopic simulations scope single vehicles but not their sub-structures and mesoscopic simulations are made up from a mix of microscopic and macroscopic models.

Closed loop vs. open loop: A simulation run can be conducted in an open loop or a closed loop manner. Within open loop simulations, the system under test is affected by its surrounding, however the surrounding does not react to the system's actions. In closed loop simulations, both the vehicle and its surrounding affect each other in both directions. One particular instantiation of open loop simulations is the augmented replay of measured data [131]. Since in reality an automated driving function will heavily influence its surrounding it is preferable to use closed loop simulations.

Inclusion of reality in simulations: It can be quite demanding to model the test object of a simulation with proper accuracy. Hence, the inclusion of parts (hardware/software) of real vehicles in the simulation run can be highly beneficial. This approach in general is denoted as X in the loop (XiL). Within X in the loop methods, parts of the models in the simulation run are replaced by reality. As an example, it is possible to replace parts of the automated driving function in the test object by real software (software in the loop; SiL). If some real hardware components (e.g. software running on a real control unit) is included, the approach is usually described as Hardware in the loop (HiL) testing. The replacement of model components by real software and hardware can be applied on different scales. An extreme example is the Vehicle Hardware in the Loop

(VeHiL) technique. Here, the whole test object (the entire vehicle which is controlled by the ADF) and even nearby surrounding vehicles are replaced by real hardware [34]. Generally speaking, the more model components are replaced by real implementations, the more accurate the simulation results will become. However, the simulation speed will decrease with an increasing share of replaced models. For VeHiL it will even be limited to real time.

3.2.2.2 Test object

In principle, the test object can be described by white-box and black-box models. A white-box model allows insights into internal states of the test object, a black-box model does not. A test object model contains several modular components. Firstly, models for the sensors used by the tested driving function must be modeled. These models can become quite complicated depending on their accuracy. Physical low-level models and phenomenological high-level models exist [98]. It is also possible to utilize sensor hardware in an X in the loop manner. Additionally, models for electronic control units and models for the vehicle dynamics such as steering and braking are needed. Especially for the modeling of the vehicle dynamics, a wide variety of models has been developed. There are simple single-track models and more advanced and more accurate and complicated multi-track ones [70]. Last but not least, it is of course very important to represent the functional chain of the tested ADF itself. This can be realized by models or again by the use of software and/or hardware components. Since the development of accurate test object models is not in the scope of this work, simple rule-based mathematical models will be used for the test object's description in the remainder.

3.2.3 Evaluation/test metrics

For the evaluation of simulation runs, it is necessary to assess their criticality. Therefore, safety metrics must be defined. Helmer [42] has introduced a classification for metrics which consists of two axes: A first distinction between *microscopic* and *macroscopic* metrics and a second distinction between metrics working only on accident cases or also on non-accidents as shown in table 3.1. Microscopic metrics evaluate single simulation runs, whereas macroscopic metrics determine statistical results regarding multiple runs. These statistical results usually can be expressed by occurrence rates

Table 3.1 Specification of safety metrics based on [42], p.53. Table was already published by Jesenski et al. [133], © 2019 IEEE.

	Metrics for accident severity	**Metrics for criticality**
microscopic	▪ physical metrics (collision speed, delta of velocity (DV) [77], ...) ▪ physiological metrics (injury severity [127], ...) ▪ economic metrics	▪ physical metrics (PET [6, 77] , WTTC [113], TTX (TTC [40], TTB [44], ...), time headway, DCE [26], more metrics surveyed by Mahmud et al. [69] ...)
macroscopic	**Metrics on accident cases** ▪ injury rate [127] ▪ fatality rate ▪ ...	**Metrics on all cases** ▪ accident rate [130] ▪ prevention rate ▪ conflict rate [127] ▪ ...

of different types of events. Microscopic metrics concerned with accidents can work on three levels: They can assess physical properties of an accident, they can have a look on the physiological effects the accident has on the passenger or pedestrians involved in the accident and it is of course possible to have a look at monetary/financial aspects of an accident. In contrast, metrics which evaluate the criticality of non-accident events without a collision can only be assessed with regard to physical measures. For instance, microscopic measures to evaluate the criticality of a simulation run can be defined by the post encroachment time (PET), several time-to-X (TTX) metrics like time-to-collision (TTC) or time-to-brake (TTB), the worst-time-to-collision (WTTC) and the time headway. This list of metrics is of course a long way from being complete. An extensive study discussing different metrics was for example realized by Mahmud et al. [69]. In addition to these deterministic metrics, probabilistic metrics have been developed. As an example, a risk-based framework which generalizes the

TTC to a time-to-closest-encounter (TTCE) which measures the time to a distance-of closest-encounter (DCE) is given by Eggert [26]. In addition to metrics which purely assess the safety implications of automated driving functions, metrics which evaluate other types of effects might be beneficial and necessary. As an example, the application of traffic quality metrics as discussed in [39] might be advantageous.

All the previously mentioned microscopic metrics are united in having the same problem: They evaluate the criticality of a scenario, but they do not allow assessing which traffic participant is to blame for the causation of the same. These metrics shall therefore be called to be purely *physics-based* metrics. The question of blame is essential since the introduction of ADFs will not be able to prevent the occurence of all accidents since the world is complex and a certain number of accidents caused by environment vehicles will affect the ADF-vehicle even if its behavior is ideal. Therefore, *behavior-based metrics* assessing the blame are required. A model for responsibility-sensitive safety (RSS) was published recently [97]. The creators of RSS define *minimal safe lateral* and *longitudinal distances* between two vehicles. These safe distances are based upon dynamical worst case assumptions. In the case of an undercut of the distances, RSS demands the affected vehicles to execute certain *proper responses*. The failure to react according to these pre-defined rules results in the assignment of blame for a possibly happening accident to the misbehaving vehicle. Although RSS defines minimal safe distances and requires vehicles to handle them in a certain way, it does not directly introduce qualitative metrics which can be used for the assessment of safety. Hence, Arechiga [9] and Hekmatnejad et al. [41] translate RSS into a formal signal temporal logic (STL) description. Besides, the authors discuss the use of a formal approach to STL robustness [29] which makes it possible to automatically derive robustness metrics based on the STL descriptions. Alternatively defined RSS-based metrics are presented by Wishart et al. [123].

3.2.4 Metrics for model validation

As already mentioned in section 3.1.1, it is impossible to exactly reproduce RB. Therefore, it is of utmost importance to find good approximations of the ∞-complex context and the aimed purpose. In order to show the accuracy of these models which are usually created using large datasets, it becomes necessary to develop metrics quantifying

- the *accurateness* of the models when reproducing the relations learned from the dataset and,
- the *completeness* of the underlying dataset used for model creation.

Commonly used metrics for the evaluation of the models' accurateness are based on comparisons between distributions in the dataset and distributions in the scenarios sampled by the learned models. These comparisons can scope "emergent" behavior parameters [64, 119, 120] or parameters directly learned from the dataset. Mathematically, the comparisons can be executed using for example Kullback-Leibler divergence (KBL) [64, 74], cross-validated likelihoods [120], root-mean-square error (RMSE) [35, 65] and mean-absolute error (MAE) [65].

In contrast to comparisons applicable to the assessment of accuracy, metrics evaluating the completeness of a dataset must directly work on the dataset itself. Hence, the development of such metrics requires a good theoretical and methodological foundation. For instance, Gelder et al. [31] use the mean-integrated-squared error (MISE) to measure the quality of distributions estimated from a dataset of limited size which then can be used to derive information about the "completeness" of the dataset.

3.3 State of the art in comparison to performed research

As already discussed in chapter 1 and reflected in the contributions listed in section 1.3, this thesis comprises three methodological main concepts: The creation of a scene model, work on methods for IS and the proposition of a new behavior-based safety metric. In the following, it shall be analyzed how these new concepts compare to the current state of the art.

3.3.1 Scene generation

The PGM-based scene model in this work's scope shall be able to statistically create scenes without the limitation on distinct maneuvers. Hence, the aimed model must belong to the generic modeling approach as discussed in section 3.2.1.2. Wheeler [117, p. 86-90] categorizes statistical generic scene generation approaches into the classes of *Scene Database* (section 3.3.1.1),

Generation via Simulation (section 3.3.1.3), *Incremental Roadway Population* (section 3.3.1.5) and *Factor Graph Models* (section 3.3.1.4). In addition to Wheeler's taxonomy, *knowledge-based* (section 3.3.1.2) scene creation is considered in the following.

3.3.1.1 Scene database (data-driven)

The most obvious way to generate the initial scene for a simulation run is by drawing it from a dataset. This approach is problematic since the required size of a dataset needed to represent a test space quickly explodes with an increasing number of scene parameters [117, p.87]. To give a lower boundary to the extent of the needed dataset, it is possible to approximate the combinatorics when considering the dynamical traffic participants in the 3. layer of Schuldt's 4-layer model [95, p. 7]. For an estimation of the lower boundary, let every dynamical traffic participant be fully specified by the four parameters of lateral position, longitudinal position, velocity and heading angle. Then the combinatorics N of the complete scene description for n vehicles is given by

$$N = d^{4 \cdot n}, \tag{3.1}$$

if it is further assumed that d is a sufficiently large number of discrete levels in the parameter ranges. By way of example, for $d = 20$ and $n = 6$ the number of possible scenes already increases to $N \approx 1.68 \times 10^{31}$. When considering all 4 layers and needing a larger number of discrete levels per parameter range, this number will increase dramatically. Hence, it is unlikely that databases with a reasonable size will ever be able to comprehensively represent the statistics of even simple scenes. In addition, a scene database is not capable of giving the global likelihood of a scene as required by methods like IS. It is also not possible to optimize on critical samples without adding a model-based approach to the dataset.

3.3.1.2 Knowledge-based scene generation

Another possibility to generate scenes is based on the usage of expert knowledge. However, due to the open context environment ADFs operate in, expert knowledge runs into the same curse of dimensionality as the database approach. Additionally, a knowledge-based approach without the use of data-driven methods does not allow to extract the statistics of scene parameters. However, expert knowledge can definitely support

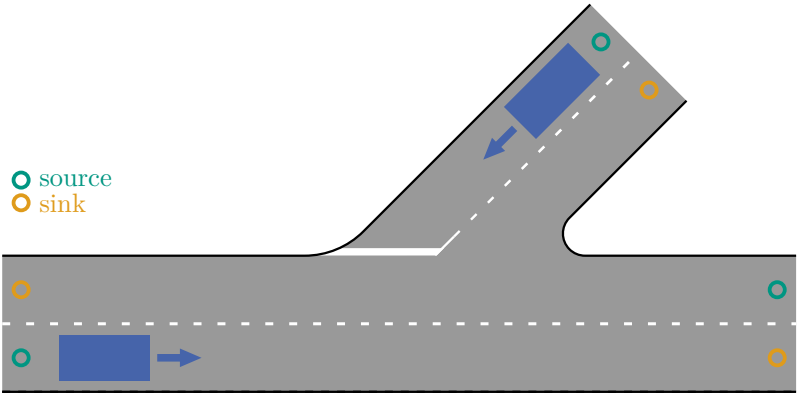


Figure 3.8 A scene is generated via simulation. Vehicles spawn at the sources and vanish at the sinks. During the burn-in time the vehicles drive into the topology according to behavior models. Afterwards the created scene is said to statistically represent the real scene distribution.

the development and release of ADFs by identifying and testing certain known-to-be-critical scene types.

3.3.1.3 Generation via simulation

It is also possible to generate scenes by spawning vehicles at certain points (*sources*) and deleting them at different points (*sinks*) in the road topology. Figure 3.8 illustrates the approach. As also discussed by Wheeler [117, p. 88], this method requires a burn-in time which allows the vehicles to dispense on the whole road topology. They accomplish that by applying behavior models which define their dynamics. In principle, the source positions of the vehicles can be chosen as desired, but often positions at the border of the road topology are chosen. The application of generation via simulation does not only require fine-granular and well performing behavior models, but also a good statistical representation of the vehicles to be spawned at the source locations. Additionally, it is of course necessary to generate the *intents* of the vehicles. That is, the behavior models must know which endpoints of the road topology a vehicle wants to reach. This information then can be used to calculate a route for each vehicle.

The simulation tool SUMO [67] for example uses this scene generation method. SUMO hereby offers different tools to determine the *flows* of

vehicles entering at the sources and leaving at the sinks. Usually SUMO uses rather simple distributions, e.g. the probability of vehicles entering the road topology at a distinct source can be modeled by a binomial distribution. Speed distributions can be modeled by truncated Gaussian distributions. SUMO also provides tools to generate the intents/routes of the vehicles. For instance SUMO is able to automatically determine them by only being given the flow rates at the sources and sinks of a specific road topology. For more information see also SUMO's online documentation [100]. Wheeler [117, p. 88] further notes that tools like MITSIM and PreScan use similar spawning methods as SUMO.

Since SUMO focuses on the modeling of large road networks with even hundreds of vehicles and not on modeling the environment of a single ADF as accurate as possible, these procedures seem to be appropriate. However, when a highly accurate scene description is required the used simple approaches might run into certain limitations.

The described approach is challenged by the necessity of finding good locations for sinks and sources and the difficulty to create good performing statistical models for the spawning at each source. For the sake of complexity, distribution-free models would be advantageous. Besides, it would be highly beneficial to learn models which condition the distributions of the spawning parameters of new vehicles on properties of already spawned vehicles and on conditions of the road topology. Assume for example that the road topology gets jammed within the burn-in time. If the spawning at the sources does not acknowledge that, vehicles might be spawned at high velocities and directly crash into the end of the traffic jam at a very high frequency. The statistics of the spawned vehicles then are obviously not realistic and non-representative scenes might be created. However, if a highly detailed and conditional modeling of vehicle spawning statistics becomes necessary, other methods which also need such models – e.g. see section 3.3.1.5 – and which do not require a burn-in time, vehicle intention models or behavior models, should be a better choice.

Depending on the size of the simulated road topology, the burn-in time could potentially become very large since it is necessary to wait at least for the first vehicles to completely transit the whole topology. The needed burn-in time might even be longer since the first vehicles driving in the roads might not behave fully consistent since they do not have any predecessors which influence their dynamics (e.g. they do not have to wait at fully populated intersections).

Another problem is caused by the inaccuracies included in behavior models. For increasing predicted time periods, the deficiencies of the behavior models should stronger affect the statistical representations. The reason is their sequential application which might lead to the accumulation of deviations. Similarly, the creation of intention models can be challenging and needs high-level data about the whole road topology.

Last but not least, depending on the accuracy and conditionality of the statistical spawning models, it might never be possible to reach good scene representations at the borders of the topology near to the sources. This is caused by the newly spawned vehicles not having enough time to reach a good statistical state near the borders. Especially for small road networks, this might invalidate a large proportion of the created scenes.

As already discussed, the simulation-based validation of ADFs will be highly inefficient if the statistical sampling methods do not emphasize critical parameter ranges/critical scenarios. This adds to the hurdles of the generation via simulation approach, since this approach does not allow to push certain parts/vehicles of the final scene (e.g. vehicles which relate in a certain way with the ADF under test) into critical states since at the sampling time the end positions of the vehicles are not (fully) known yet. In addition, the approach is not capable of giving global likelihoods for a generated scene (since multiple generation via simulation paths might lead to the same resulting scene).

3.3.1.4 Factor graph models

Wheeler and Kochenderfer [118] recently published a statistical scene generation method based on factor graphs. This method augments initial scenes drawn from a dataset to statistically represent learned distributions. For that, the joint probability distribution $p(\varphi_{\boldsymbol{x}_0})$ – with $\boldsymbol{x}_0$ describing the set of parameters necessary to define a scene – is factorized by

$$p\left(\varphi_{\boldsymbol{x}_0}\right) = \frac{1}{Z} \prod_i \mathcal{F}_i^{type_i} \tag{3.2}$$

into a set of factors $\left\{\mathcal{F}_i^{type_i}\right\}$. φ_X denotes a numerical value assigned to the random variable(s) in X. This set of factors contains elements of different types such as factors considering the relations of a vehicle to following and neighboring vehicles or to the lane itself. The factorization is illustrated in fig. 3.9. Each of these factors' scope consists of a subset of scene parameters.

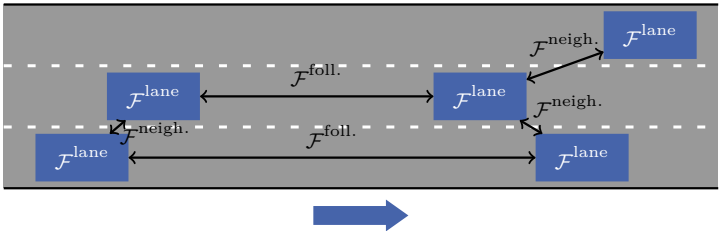


Figure 3.9 Illustration of factor graphs. The total joint probability distribution is factorized into factors considering the relation of a vehicle to the road topology ($\mathcal{F}^{\text{lane}}$), to the predecessor/successor in its own lane ($\mathcal{F}^{\text{foll.}}$) and to the nearest vehicles in the neighbouring lanes ($\mathcal{F}^{\text{neigh.}}$).

Wheeler and Kochenderfer [118] for example define the following-factor $\mathcal{F}^{\text{foll.}}$ to be dependent on the relative speed Δv and the distance headway d between consecutive vehicles. They fit the factor graph distributions to a dataset by a gradient-descent-based procedure.

The scene generation always starts by drawing a scene from the dataset. MCMC methods like the Metropolis-Hastings algorithm are then used to statistically augment this drawn initial scene by formulating the augmentation procedure as a Markov chain (the drawn initial scene is the first element in the chain). Note, that for each step in the Markov chain, one vehicle in the scene is selected with uniform probability, updated as described by a *transition* distribution and this update is then accepted or rejected as described by an *acceptance ratio* which is dependent on p (see algorithm 1, [118]). After a sufficiently high number of burn-in steps – that means after a Markov chain of sufficient length – the scenes as elements of the Markov chain change into a stationary distribution representing the learned statistics $p\left(\varphi_{\boldsymbol{x}_0}\right)$.

On the one hand, factor graphs have the strength, that they are quite flexible regarding the topology of the modeled road. It is straight forward to add new factors and to model different types of correlations. Additionally, the factor graph approach allows to fuse information in a dataset and information inherent to a learned model. Wheeler [117, p. 115] argues, that the usage of scenes drawn from a dataset as initial elements of the Markov chain ensures realistic roadway populations. Wheeler additionally shows that factor graphs can well reproduce a dataset's marginal parameter distributions.

On the other hand, there are major challenges inherent to this approach. Firstly, a quite large burn-in time for the MCMC algorithm may be needed. Wheeler and Kochenderfer [118] use 1000 steps. Secondly, it's not possible to modify or adapt a learned road topology since a dataset to draw the initial scenes from is required. In this connection, for the sampling it is also required to handle a possibly very large dataset even after having trained the model. Thirdly, it is not possible to use this approach in order to calculate a global likelihood for a scene [117, p. 116]. The model only includes local likelihoods within scenes generated from the same initially drawn dataset scene. As it will become clear later, this causes problems regarding the application of other methods like IS which require knowledge of global likelihoods. With regard to IS, the absence of global likelihoods makes it necessary to introduce a clustering of scenes which includes major flaws as discussed later in section 3.3.2. Fourthly, the method requires the manual selection of features and parametric shapes for the factors of the probability distribution. This limits the expressiveness of the whole model. Besides, dependent on the chosen features in the single factors, certain independence assumptions between parameters are automatically included. For instance, the following-factors between the first, second and third vehicle in a lane have the form of $\mathcal{F}^{\text{foll.}}(\varphi_{d_{1,2}}, \varphi_{\Delta v_{1,2}}) \cdot \mathcal{F}^{\text{foll.}}(\varphi_{d_{2,3}}, \varphi_{\Delta v_{2,3}})$. This results in $(d_{1,2}, \Delta v_{1,2}) \perp\!\!\!\perp (d_{2,3}, \Delta v_{2,3})$. Fifthly, it has not been shown yet, that it is really possible to apply the approach to arbitrary road topologies. It seems questionable that the current state of the approach works for inhomogeneous topologies like intersections since there exist different statistic correlations at different locations within a road topology. Therefore, the assumption that the factors' shape is the same everywhere should cause problems. By way of example, it is not clear how to apply the approach in the surrounding of a stopping line in an intersection. Last but not least, Wheeler [117, p. 112-116] mentions that depending on the size of the dataset, the scenes modeled by the factor graph approach might not be able to completely represent all possible scenes. This is rooted in the fact that certain properties like number of vehicles, vehicle widths, lengths, orderings etc. are always set by the drawn initial scene and cannot be augmented by the MCMC procedure. Hence, it might be beneficial to improve the factor graph model by drawing the initial scenes using an incremental roadway population model as discussed in the next chapter.

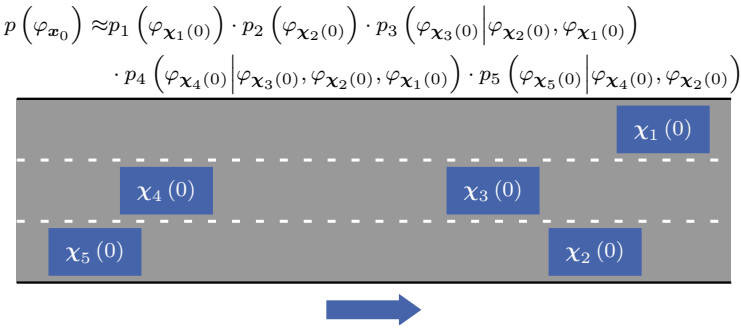


Figure 3.10 Illustration of incremental roadway population. In this example, the vehicles are conditioned on the predecessor in the same lane and on the predecessors in the neighboring lanes. The set $\boldsymbol{\chi}_i\,(0)$ describes all parameters necessary to completely define vehicle i. The vehicles are then generated incrementally by sampling $\boldsymbol{\chi}_i\,(0) \sim p_i$.

3.3.1.5 Incremental roadway population

Another possibility to statistically model scenes are incremental roadway population models. These models, similar to factor graphs, work by splitting up the joint distribution of scene parameters into several factors by utilizing certain independency assumptions. However, incremental roadway population models enable modeling with conditional distributions which allow considering important correlations between vehicles.

Rule-based sampling order: The independency assumptions for incremental roadway population models can be based on the scene configuration/the road topology. This dependency on the road topology shall be expressed by the term *rule-based* sampling. An exemplary way to factorize the total probability distribution in a rule-based fashion is shown in fig. 3.10. During rule-based sampling one lane after another is sampled which allows setting vehicles' properties in relation to the lane and to predecessors and successors. To the author's best knowledge, the method has previously only been applied to straight highway sections [119]. Significant methodological improvements are necessary to apply it to more complex topologies.

In comparison to the factor graph approach, the incremental roadway population method has some weaknesses and some strengths. On the side of the strengths, no burn-in time is required. It is further possible to sample in changed road topologies as long as all required types of correlations

have already been learned by the conditional probability distributions. For instance, if the length of the straight highway displayed in fig. 3.10 was doubled it still could be populated as long as the correlations remain the same throughout the whole length (that means, the conditional dependence on the first predecessor in the same lane and on the neighbor lanes). Consequently, when having learned all necessary correlations it should be possible to combine them into new topologies which are modifiable by a certain degree. When using suitable distribution-free models, it should additionally be possible to learn the conditional distribution without being limited to a distinct functional shape. In contrast to the factor graph model, the incremental roadway population does not only allow the generation of initial scenes for a simulation, but should be further applicable to sample new vehicles which enter the simulated topology during the dynamics part of the simulation. That means, properly learned conditional distributions could in principle also be used as the statistical models required at the sources for the approach discussed in section 3.3.1.3. As another advantage, after having trained the models, the dataset is not needed anymore for scene generation. Furthermore, the incremental approach allows to determine global likelihoods for each scene which is important for the application of certain statistical simulation methods. In this connection, it is also possible to modify specific parts of a scene which are in particular critical for an ADF-vehicle since it is already known at sampling time which vehicles are located next to the ADF-vehicle.

The greatest weakness of this rule-based approach is that it strongly relies on the concepts of predecessor and successor which correlate to the definition of lanes. Therefore, with regard to the road topology type this approach is not as flexible as the factor graph method since the splitting of the joint probability distribution requires to consider the different types of relations between different lanes. In this context, it is straight forward to condition the probability distributions on vehicles positioned in the same lane as the vehicle which is sampled [119]. However as it is shown in section 6.2, it is more complicated to apply this method to topologies which include splitting, merging and intersecting lanes. Also the modeling of neighboring lanes requires further effort. In particular, it may also be challenging to model scenes which include multiple vehicles located laterally next to each other in the same lane, e.g. a parking vehicle located in a certain lane which is bypassed by another vehicle in the very same lane. Another challenge is the modeling of pedestrians since the concept of lanes

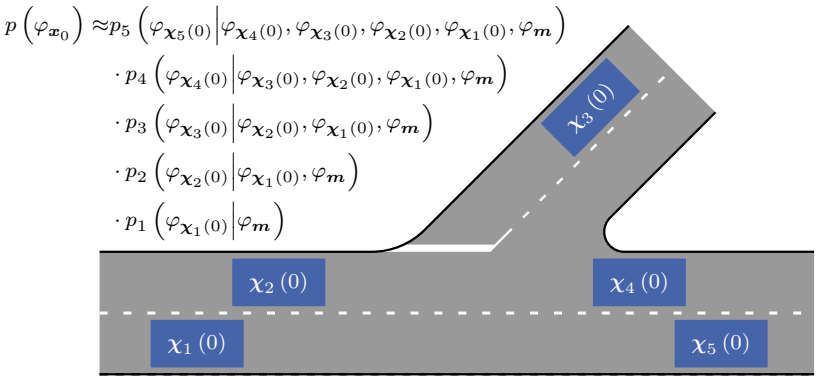


Figure 3.11 Alternative incremental roadway population approach as introduced in [102]. In this example, the vehicles are conditioned on the vehicles which were previously sampled in a left to right, top to bottom ordering. Again, the set $\boldsymbol{\chi}_i\,(0)$ describes all parameters necessary to completely define vehicle i. $\boldsymbol{m}$ includes all the information required to completely describe the road topology. The vehicles are generated incrementally by sampling $\boldsymbol{\chi}_i\,(0) \sim p_i$.

does not necessarily apply to them.

In summary, there are several arguments which render this method to be the best choice for the purpose of this thesis: Firstly, the incremental approach can model the global likelihood of a scene. Secondly, since it is already known at sampling time which vehicles are located next to the ADF-vehicle, it is possible to bias them to be critical. Thirdly, the absence of a burn-in time is beneficial. For the model to be useful, it is however necessary to show its applicability on more general road topologies than the straight highway sections implemented by Wheeler et al. [119] (section 6.2), to increase the efficiency by reducing the sampling time (section 6.3) and to show how to use it to create predominantly critical scenes (section 7.1).

Rule-free sampling order: For the sake of completeness it shall be mentioned, that after the experimental work in this thesis was performed, Tan et al. [102] proposed in 2021 an alternative sampling order in order to overcome the strict dependency on the concepts of predecessor, successor and lane. In their work, a scene was not incrementally filled by following lanes, as illustrated in fig. 3.10, but *rule-free* (that means the sampling order is independent from the road topology) from the left to right, top to

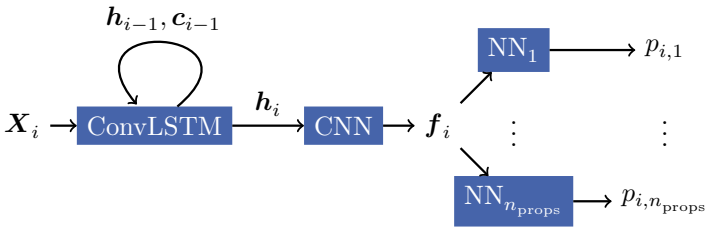


Figure 3.12 Rough neuronal network structure used to model $p_i\left(\varphi_{\boldsymbol{\chi}_i(0)} \middle| \varphi_{\boldsymbol{\chi}_1(0)} \cdots \varphi_{\boldsymbol{\chi}_{i-1}(0)}, \varphi_{\boldsymbol{m}}\right)$. NN is short for neuronal network.

the bottom in the considered road topology as shown in fig. 3.11. Since this alternative sampling order does not implicitly consider the road topology $\boldsymbol{m}$, it becomes necessary to explicitly condition on it. Tan et al. [102] modeled the required probability distributions

$$p_i\left(\varphi_{\boldsymbol{\chi}_i(0)} \middle| \varphi_{\boldsymbol{\chi}_1(0)} \cdots \varphi_{\boldsymbol{\chi}_{i-1}(0)}, \varphi_{\boldsymbol{m}}\right) = \prod_{k=1}^{n_{\mathrm{props}}} p_{i,k}\left(\varphi_{\boldsymbol{\chi}_{i,k}(0)} \middle| \varphi_{\boldsymbol{\chi}_1(0)} \cdots \varphi_{\boldsymbol{\chi}_{i-1}(0)}, \varphi_{\boldsymbol{\chi}_{i,<k}(0)}, \varphi_{\boldsymbol{m}}\right) \tag{3.3}$$

which describe the next sampling step i by a neuronal network architecture. This neuronal network architecture is roughly given in fig. 3.12. Variable $\boldsymbol{\chi}_{i,k}(0)$ gives the kth property of the vehicle sampled in the ith iteration. $\boldsymbol{\chi}_{i,<k}(0)$ gives the first $k-1$th properties and $p_{i,k}$ is used to sample the kth property. A convolutional LSTM (ConvLSTM) network was used as an entrance point for the iterative sampling. For each sampling step i, a new multi-channel image $\boldsymbol{X}_i$ which describes the already sampled vehicles $\boldsymbol{\chi}_1(0) \cdots \boldsymbol{\chi}_{i-1}(0)$ and the road topology $\boldsymbol{m}$ is input into ConvLSTM. In order to do so, each property of the road topology and of the already sampled traffic participants is encoded within one of the channels of $\boldsymbol{X}_i$ (the complete region of interest of the scene is discretized and the pixels are assigned the properties at a certain point within the scene). The hidden layer output $\boldsymbol{h}_i$ of ConvLSTM is then delivered to a convolutional neuronal network (CNN) which extracts some baseline features $\boldsymbol{f}_i$. The baseline features contain information about the current scene and all vehicles which have already been sampled. Since ConvLSTM remembers about the iterations $1 \ldots i-1$ via the recurrent usage of cell state $\boldsymbol{c}_{i-1}$ and hidden state $\boldsymbol{h}_{i-1}$ of

iteration $i-1$, information from the complete sampling procedure will be summarized in $\boldsymbol{f}_i$. The features $\boldsymbol{f}_i$ are then used as an input to several neuronal networks which predict the parameterization of the probability distributions $p_{i,k}$ which are necessary to sample the next vehicle.

Since this sampling order does not care about lanes, it is more flexible and can be used to sample lane-independent traffic participants like pedestrians. However, there are certain disadvantages.

Firstly, this kind of rule-free sampling induces a high possibility to create invalid scenes. For example, vehicles can easily be sampled at positions which cannot be accessed by them, vehicles can be sampled overlapping with already existing traffic participants and so on[1]. Tan et al. [102] prevented such cases by manually adapting the learned probability distributions after the training (e.g. not allowed locations are manually set to zero) and by rejecting the sampling of vehicle who collide with already existing vehicles. In addition, for each iteration i they sampled multiple times (10 times) and only accepted the sample with the highest likelihood to get rid of invalid outliers. Obviously, these measures change the learned probability distribution especially in its "tail" section. However, when using scene generation models for validation, especially these tails may be of great interest since the interesting scenes which challenge an ADF usually occur quite seldomly. They are therefore located in the tails of the distributions. Hence, a skew in the distribution's tail may have strong negative impacts on validation. When considering lanes by applying rule-based sampling, these problems are automatically solved since the consideration of the road topology during the sampling procedure and the sampling along a lane inherently prevents degenerate scenes.

Secondly, it is quite hard to model the conditional part of the conditional distributions $p_{i,k}$ with this neuronal network architecture. Tan et al. [102] try to mitigate this problem e.g. by using different networks for different vehicle classes and by scoping the baseline feature vector to the respective pixel where the vehicle is sampled. However, this measures do not scale and cannot solve the problem in general. E.g. Tan et al. [102] had to ignore the conditioning on the vehicles' bounding boxes when sampling a vehicle's velocity. In the future, when even more vehicle properties might be needed the problem will become even more prominent.

[1] of course the conditioning on the map and already existing vehicles should lower the probability for such degenerate cases, but the learned distributions typically have long tails

Thirdly, in reality lane-dependent traffic participants will always adapt their behavior based on surrounding traffic participants which relate to them in a certain rule-based fashion. The rule-based sampling approach allows the explicit training/fitting of the distributions on these relations which are most relevant for the safety of an ADF by cleverly selecting the random variables[2] used to build up the scene. Consequently, the explicit training of these safety-relevant, rule-based relations should be more precise than if the relations are implicitly obtained when sampling rule-free[3].

Fourthly, rule-based sampling further allows finding good statistical independencies between traffic participants and random variables when modeling the joint probability distribution, whereas during rule-free sampling, the distributions must be conditioned on all previously sampled vehicles (see fig. 3.11).

Fifthly, statistical methods as IS necessitate the sampling of predominantly critical scenes. In this context, it is of great benefit to use some kind of safety metric to assess scenes (see section 7.1). However, such safety metrics usually consider some road-topology based *rules* [4]. That makes it much easier to apply safety metrics to the rule-based sampling procedure, which will be illustrated in section 7.1. Note also, that no method to statistically sample predominantly critical scenes was given in [102].

Sixthly, in order to implement the rule-free sampling procedure, one still must completely understand the underlying traffic rules within a road topology since they must be encoded in the respective channels of $\boldsymbol{X}_i$. Therefore, the effort to implement the sampling of lane-base traffic participants on complex road topologies will be similar to the effort of applying the generalized rule-based sampling procedure developed in section 6.2.

As a conclusion, the main advantage of this new, rule-free sampling procedure is its capability to sample lane-independent traffic participants like pedestrians. For these cases it is however easily possible to first sample the lane-based traffic participants in the rule-based manner and sample the pedestrians rule-free afterwards. For that, the properties $\boldsymbol{m}_{lbTP}$ of

[2] e.g. enable explicit training by clever selection of directly sampled random variables like distance to predecessor when sampling a lane in rule-based fashion. When sampling rule-free one would have to use a more generic variable like an absolute position during the sampling since there would be no predecessor one can refer to

[3] direct training of distance between two consecutive vehicles vs training of two absolute positions which will implicitly result in a certain distance.

[4] or more precisely: lane-dependent relations between vehicles

lane-dependent traffic participants can just be added in the conditional part of

$$p_i \left(\varphi_{\boldsymbol{\chi}_{\text{ped.},i}(0)} \middle| \varphi_{\boldsymbol{\chi}_{\text{ped.},1}(0)} \cdots \varphi_{\boldsymbol{\chi}_{\text{ped.},i-1}(0)}, \varphi_{\boldsymbol{m}}, \varphi_{\boldsymbol{m}_{lbTP}} \right). \tag{3.4}$$

3.3.2 Importance sampling

As already discussed in section 3.2.1.1, methods to enhance the efficiency of statistical sampling methods are highly desirable. In the framework developed in this thesis, importance sampling is applied for this purpose. Due to the scaling issues discussed in section 1.3, recent work on IS is limited to scenarios of restricted complexity or a specific maneuver type. Exemplarily, IS was applied to maneuver-based models for lane change [126], lane following [30, 127] and for pedestrian crosswalk scenarios [116].

O'Kelly et al. [80] generalize the application of IS to highway scenarios with surrounding vehicles governed by a generic behavior model based on generative adversarial imitation learning (compare section 3.2.1.2 for a classification). However, the scenario is limited to highway scenes with five environment vehicles, the parameter space of the behavior model is strongly restricted for feasibility and all the environment vehicles are controlled by the same parametrization of the behavior model. Especially the last fact is a limitation to IS since critical behavior of different environment vehicles should differentiate a lot. For example, the predecessor of the ADF vehicle could brake strongly, whereas the successor could do the opposite and drive as fast as possible in order to challenge the ADF. The authors refine their framework by applying subset simulation methods to a real commercial AV system [78]. Subset sampling has the advantage, that – in contrast to IS – no assumptions about the parametric form of the distribution describing critical events are required. However, subset sampling does not natively describe a distribution of critical scenes and therefore it is harder to get more insights into the system's failure regions. Additionally, the work done by Norden et al. [78] still shows the same limitations regarding the scenario as the original work [80], as already mentioned before.

Recently, Wheeler and Kochenderfer [121] implemented importance sampling based on factor graphs and clustering of critical scenes which should in principle be scalable to scenarios of higher complexity. The approach works as follows: Firstly, scenarios are drawn by MCS from the unbiased scenario distribution. Secondly, scenarios ending in a collision within this unbiased sampling are detected and recorded. These collision scenarios can

be used to find scenarios with a certain collision criticalitiy/probability. This is achieved by conducting numerous simulations/finite-horizon MCS runs starting at scenes at different time-steps before the collision. The criticality of a scene at a certain time step in the forerun of the collision is then given by the share of simulation runs started at this time step which result in a collision. The scenarios/scenes at time steps which were found to exceed a defined criticality value are automatically clustered into collision classes with different collision causes. Thirdly, factor graphs are learned to represent the scene distributions for each of these critical scenario clusters. Fourthly, by stochastically sampling scenarios from these clusters it is now possible to deduct an importance sampling distribution which should enable the accelerated calculation of statistical evidence about the system. However, this approach has a major drawback: The detection of the clusters and the MCS-based generation of a sufficient number of critical cluster scenarios requires a very high amount of simulation runs. Since the IS distribution must be properly represented, the costs should be similar to naive MCS-based generation of statistical evidence. If the created clustering and factor graphs could be reused multiple times for different ADFs and development states of an ADF, the approach could still be beneficial. However, due to the great influence of active ADFs on their surrounding, the shape of the critical scenario distributions will be strongly dependent on the tested ADF and its development progress. Therefore, calculated clusters and factor graphs cannot be reused for different ADFs or development states. Wheeler [117, p. 167] confirms this by stating, that "it is important to ensure that the optimized candidate safety system does not affect the critical clusters. Introducing an active safety system may change the frequency of critical scenarios, or may change the critical scenario categories altogether. The autonomous emergency braking system typically activates after the risk exceeds the critical transition threshold, and thus it is unlikely that the critical situation clusters would change if generated with simulations involving vehicles equipped with AEBs." Consequently, the high cost factor graph generation process would have to be repeated for each active driving function's development state and the approach seems to be limited to the validation of reactive ADFs like an automated emergency braking function.

To overcome the discussed IS scalability problem and the mentioned limitations, this thesis proposes a framework which allows applying IS selectively only onto the parts of a scenario which are relevant for the

safety/criticality of the ADF, e.g. a vehicle 400 m behind the ADF-vehicle should not be very relevant for safety and therefore should not be included in the IS optimization. Depending on the share of the relevant parts of the scenario, the resulting IS optimization problem may be (strongly) reduced (section 5.2).

3.3.3 Behavior-based metric

Section 3.2.3 discusses the necessity and different types of metrics to evaluate the outcome of a simulation run. As it was also discussed, behavior-based metrics based on STL formulations of RSS recently were introduced in order to mitigate the problem of not being able to evaluate the assignment of blame [9, 41]. The usage of STL allows the automatic derivation of metrics based on formal robustness definitions [29].

In general, the STL semantics operate on traces $s(t)$ which give the state s of a system at time t. The most basic STL element is a statement of type $\phi = (f(s(t)) > c)$ with robustness $\rho(\phi) = f(s(t)) - c$. Conjunctions of several statements can be expressed by AND and OR operations ($\wedge/\vee$). The robustness values are calculated by $\rho(\phi_1 \vee \phi_2) = \max(\rho(\phi_1), \rho(\phi_2))$ and $\rho(\phi_1 \wedge \phi_2) = \min(\rho(\phi_1), \rho(\phi_2))$. This leads to a first point of criticism: If the physical dimensions of ϕ_1 and ϕ_2 are not the same, e.g. the STL formulation of RSS contains conjunctions of m and $\frac{\mathrm{m}}{\mathrm{s}^2}$, this is no natural applicable relation, similar to a possible comparison of units km and m. Analogously, the operator $\square_I \phi$, which states that the statement ϕ is valid during the whole interval I, is as robust as given by $\rho(\square\phi) = \min_{t \in I} \phi$. That implies a second point of criticism: The minimization does only consider the time point with the worst robustness to calculate the total robustness of a whole interval. In contrast, in reality the probability of an accident will dependent significantly on the length of the time span in which robustness is low. Take for example the attentiveness of a driver. The likelihood to cause an accident will be way smaller for a driver who is inattentive for a fraction of a second than for a driver who is not watching the vehicle's surrounding for several seconds. As a third point of criticism it is hard to interpret the numerical criticality values generated by the automatically generated STL robustness formulas, e.g. there is not necessarily a direct correspondence to accidents (it may for example not be clear which value corresponds to an accident or other qualitative events).

Besides, as also mentioned in section 3.2.3, Wishart et al. [123] have

introduced an RSS motivated behavior-based metric called *proper response action* (PRA). PRA analyzes if the minimal safe distance defined by RSS is undercut and if the vehicles do not show the required response demanded by RSS. If both is true, PRA returns 1, otherwise it returns 0. PRA is therefore a binary metric and not sufficient for applications such as IS (compare section 4.2) which require continuous criticality values.

In order to solve these limitations in the state of the art, section 4.4.3 uses RSS to derive a continuous behavior-based metric which does not only consider the absolute value of the criticality/robustness, but also the time span a scene was critical. Additionally the metric omits unnatural comparisons.

4 Metrics

Section 3.2.3 shows that without well-defined microscopic metrics, which evaluate the safety and other aspects of a simulation run, it would not be possible to extract sensible information of a performed simulation. Hence, this chapter centrally considers the microscopic metrics in use within the scope of this work. In detail, section 4.1 mathematically formalizes the notion of a simulation run and section 4.2 discusses the required properties and construction of microscopic metrics. Sections 4.3, 4.4.2 and 4.4.3 describe the specific metrics applied throughout the remainder. While the metrics given in sections 4.3 and 4.4.2 are conventional physics-based ones, the metric proposed in section 4.4.3 is a behavior-based metric uniquely contributed by this thesis. It is designed to satisfy the requirements in section 4.2 and to address the limitations of the current state of the art outlined in section 3.3.3. For that, section 4.4.1 recaps the basic RSS formulas.

4.1 Mathematical description of a simulation run

As already described in section 2.1, a simulation run is equivalent to a scenario and thereby is a sequence of scenes. Hence, a simulation run ξ can mathematically be expressed by the tuple

$$\varphi_\xi = \left(\varphi_{\boldsymbol{x}_0}, ..., \varphi_{\boldsymbol{x}_{n_{\varphi_\xi}}} \right) \text{ with } \varphi_\xi \in \boldsymbol{\Omega} = Val\left(\xi\right). \tag{4.1}$$

Here, $\boldsymbol{x}_t$ describes a scene at time step t. Note, that φ_ξ and $\varphi_{\boldsymbol{x}_t}$ mathematically describe full assignments of random variables ξ and $\boldsymbol{x}_t$. In chapter 5, it will become clear why they are grasped as random variables. A simulation run can have a variable time horizon (number of steps) n_{φ_ξ}. Let the time

between two time steps be given by Δt. Therefore, the scenes' timestamps are given by

$$t \in \boldsymbol{\mathcal{T}}_{\varphi_\xi} = \{0, \Delta t, ..., n_{\varphi_\xi} \cdot \Delta t\} \tag{4.2}$$

The set

$$\boldsymbol{\nu}_{\varphi_\xi} = \left\{1, ..., n_{\text{veh}}^{(\varphi_\xi)}\right\} \tag{4.3}$$

gives the vehicles involved in simulation run φ_ξ. Therefore, each assigned simulation step

$$\varphi_{\boldsymbol{x}_t} = \left\{\boldsymbol{\chi}_i\,(t) \middle| i \in \boldsymbol{\nu}_{\varphi_\xi}\right\} \tag{4.4}$$

is determined by the results of $n_{\text{veh}}^{(\varphi_\xi)}$ multidimensional functions

$$\boldsymbol{\chi}_j : \boldsymbol{\mathcal{T}}_{\varphi_\xi} \rightarrow \mathbb{R}^{n_{\text{props}}}, \tag{4.5}$$

that describe the n_{props} properties of vehicle j at time t.

4.2 Requirements and definition of microscopic metrics

A microscopic safety metric shall define a mapping of type

$$g_{\text{crit.}} : \boldsymbol{\zeta} \rightarrow \mathbb{R}, \tag{4.6}$$

with $\boldsymbol{\zeta} = \bigcup_{\varphi_\xi \in \boldsymbol{\Omega}} \left\{\left(\varphi_\xi, \boldsymbol{\mathcal{S}}\right) \middle| \boldsymbol{\mathcal{S}} \in \mathcal{P}\left(\boldsymbol{\nu}_{\varphi_\xi}\right)\right\}$. Note, that $\mathcal{P}\,(\bullet)$ determines a power set. $\boldsymbol{\mathcal{S}} \subseteq \boldsymbol{\nu}_{\varphi_\xi}$ gives the set of vehicles whose relations and criticalities shall be assessed in a simulation run. Think for example of a simulation run with only one ego vehicle which is governed by an ADF. In this case $\boldsymbol{\mathcal{S}}$ would only contain this single automated vehicle.

There are multiple requirements to the definition of the mapping $g_{\text{crit.}}$:

1. Since a different number of time steps n_{φ_ξ} and vehicles $n_{\text{veh}}^{(\varphi_\xi)}$ are possible in each simulation run, the metric must be able to operate on a set of variable size $\left|\left(\varphi_\xi, \boldsymbol{\mathcal{S}}\right)\right| = n_{\varphi_\xi} \cdot n_{\text{veh}}^{(\varphi_\xi)} \cdot n_{\text{props}} + |\boldsymbol{\mathcal{S}}|$.

2. Since the metric values will be discretized later in this work and since it is essential that the value of the criticality of different simulation runs can be compared to each other, the safety metrics should not return infinite values. Otherwise, if infinite values can happen, there must be a technical way of handling them (e.g. by mapping them to the largest occurring value in a dataset).

3. The IS approach works iteratively. Hence, the metrics are required to be continuous. A binary metric which can only distinguish between an event $\boldsymbol{\varepsilon}$ taking place ($\varphi_\xi \in \boldsymbol{\varepsilon}$) and not taking place ($\varphi_\xi \notin \boldsymbol{\varepsilon}$) will not be sufficient. For IS, the assessment of sub-criticality, that means the "distance" to the event occurring is essential (see section 5.1).

4. In a mathematical sense, $g_{\text{crit.}}$ shall define a total preorder[1] on $\boldsymbol{\zeta}$. By convention, the following discussion of $g_{\text{crit.}}\left(\xi = \varphi_\xi, \boldsymbol{\mathcal{S}}\right)$ assumes that a lower metric value corresponds to a more critical scenario. However, it is of course also possible to define metrics whose values increase with increasing criticality.

This mapping $g_{\text{crit.}}$ of tuples $\left(\varphi_\xi, \boldsymbol{\mathcal{S}}\right)$ with variable length $|\varphi_\xi| + |\boldsymbol{\mathcal{S}}|$ is realized by pairwise comparisons of the vehicles in $\boldsymbol{\mathcal{S}}$ and related vehicles. It is

$$g_{\text{crit.}}\left(\xi = \varphi_\xi, \boldsymbol{\mathcal{S}}\right) = \min_{t \in \{0,\dots,n_{\varphi_\xi}\}} \left(\min_{p \in \mathbf{A}_t^{(\varphi_\xi)}} \left(g_{\text{pair}}\left(p, t, \xi = \varphi_\xi\right)\right)\right), \quad (4.7)$$

with $\mathbf{A}_t^{(\varphi_\xi)} = \left\{ (a,b) \,|\, a \in \boldsymbol{\mathcal{S}}, b \in \boldsymbol{\nu}_{\varphi_\xi}, a \neq b, \mathcal{R}\left(a, b, \varphi_\xi, t\right) = 1 \right\}$.

$\mathcal{R}\left(a, b, \varphi_\xi, t\right)$ verifies a relation between vehicles a and b at time step t. In the simplest case $\mathcal{R}\left(\bullet, \bullet, \varphi_\xi, t\right)$ might for example check if two vehicles are consecutive vehicles in the same lane. $\mathcal{R}\left(\bullet, \bullet, \varphi_\xi, t\right)$ returns 1 if the relation is valid and otherwise 0. The pair-wise metric g_{pair} is calculated over all relevant vehicle pairs $\mathbf{A}_t^{(\varphi_\xi)}$ existent at a time step. The smallest

[1] The definition of a preorder requires the proper handling of possibly arising infinite metric values as demanded in the 2. requirement. Otherwise, reflexivity and totality cannot be guaranteed.

pairwise metric minimized over all vehicle pairs and all time steps then gives the total metric result for the entire simulation run[2].

In the next sections, possible choices for the mapping

$$g_{\text{pair}} : \bigcup_{\varphi_\xi \in \boldsymbol{\Omega}} \left\{ (p, t, \varphi_\xi) \middle| p \in \boldsymbol{\nu}_{\varphi_\xi} \times \boldsymbol{\nu}_{\varphi_\xi}, t \in \boldsymbol{\mathcal{T}}_{\varphi_\xi} \right\} \to \mathbb{R} \tag{4.8}$$

are discussed.

4.3 TTC & inverse TTC

A very popular and often used metric is the *time-to-collision* (TTC) [40]. Since TTC is used later on in this work, this chapter shall shortly give an overview over it and also shall describe how it can be incorporated in the microscopic metric approach of section 4.2. It is defined by

$$g_{\text{pair}}^{(\text{TTC})}(p, t, \varphi_\xi) = \frac{d}{v_{p_0} - v_{p_1}} \text{ for } v_{p_0} \neq v_{p_1}, \tag{4.9}$$

with $p = (p_0, p_1)$. p_0 is the successor vehicle of vehicle p_1. v_{p_1} and v_{p_0} give the velocities of the predecessor and the successor vehicle. d is their distance. Intuitively, TTC gives the time left until two consecutive vehicles would collide if they maintained their current state of movement. The situation is illustrated in fig. 4.1. TTC is very sensitive on the relative velocity $\Delta v = v_{p_0} - v_{p_1}$. In the border case $\Delta v = 0 \frac{\text{m}}{\text{s}}$, it diverges independently of the vehicles' distance d. The most critical TTC value is $\text{TTC} = 0\,\text{s}$. Negative TTC values are not critical since the predecessor vehicle drives faster than the successor vehicle in that case. Since the properties in section 4.2 require the metric to be finite and negative values are not dangerous, all negative TTC values and all infinite values found in a dataset are replaced by the largest finite TTC value found in the same dataset for the matter of ordering. The set of relevant vehicle pairs is chosen by

$$\mathbf{A}_t^{(\varphi_\xi)} = \bigcup_{i \in \boldsymbol{\mathcal{S}}} \{(i, pre[i, t]), (suc[i, t], i)\}. \tag{4.10}$$

[2] In section 3.3.3, the second point of criticism was that current metrics only consider one time point for assessing criticality, which neglects the influence of the duration of dangerous behavior. This point will be addressed in section 4.4.3 by defining a metric for $g_{\text{pair}}(p, t, \xi)$ which considers such durations.

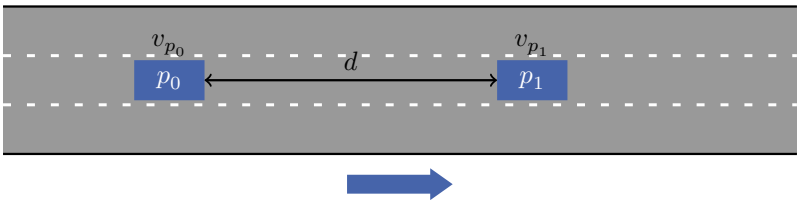


Figure 4.1 Relevant values for the calculation of TTC between the predecessing vehicle (p_1) and the successor (p_0). It is $p = (p_0, p_1) \in \mathbf{A}_t^{(\varphi_\xi)}$.

$pre[i,t]$ and $suc[i,t]$ give the predecessor respectively successor vehicle to vehicle i at time step t. Another option to handle TTC's divergence is to define the inverse TTC (invTTC) by

$$g_{\text{pair}}^{(\text{invTTC})}\left(p,t,\varphi_\xi\right) = \left(g_{\text{pair}}^{(\text{TTC})}\left(p,t,\varphi_\xi\right)\right)^{-1} = \frac{v_{p_0} - v_{p_1}}{d}. \tag{4.11}$$

invTTC has the advantage that it rarely diverges since $d = 0\,\text{m}$ is not occurring often in real-world datasets. In contrast, $\Delta v \approx 0\,\frac{\text{m}}{\text{s}}$ occurs quite often. However, metrics shall be applied to assess critical situations and since $d = 0\,\text{m}$ is a very challenging situation, invTTC diverges rarely, but at the most interesting scenarios. In order to handle diverging, infinite invTTC values during ordering, these diverging values are replaced by the largest finite values found in the same dataset. Analogously, all negative, uncritical invTTC values are replaced by the smallest positive invTTC value found in the same dataset. Please note, that deviating from the previously mentioned convention, higher, positive invTTC values are more critical than smaller positive ones. When applying (4.7) this requires the replacement of the min by max operators.

TTC and invTTC are both limited to the assessment of longitudinal vehicle pair relations. For the evaluation of lateral relations between vehicles, additional metrics are required.

4.4 RSS-based metrics

Section 3.3.3 discusses the application of RSS to define metrics which assess which vehicle is to blame for the causation of a critical situation. As also discussed there, the state of the art is limited with regard to unnatural

comparisons, the weighting of critical time spans, the formulation of a continuous metric and the interpretability. To develop a behavior-based metric which fulfills these requirements, section 4.4.1 shortly recaps RSS and section 4.4.2 gives an approach for a physics-based metric derived by RSS which can be used as foundation of the final behavior-based metric derived in section 4.4.3. Parts of this section were already published by Jesenski et al. [134]; © 2020 IEEE.

4.4.1 Responsibility-sensitive safety

In 2017, Shalev-Shwartz et al. [97] introduced responsibility-sensitive safety (RSS). RSS is a framework which does not only allow assessing the criticality of a scenario but also includes an assignment of blame should an accident occur. RSS works by the definition of lateral and longitudinal safe distances which are constructed based on physical worst case assumptions. The *minimal safe longitudinal distance* [97, p. 7] is given by

$$d_{\min}^{(\text{long})}(v_{\text{suc}}, v_{\text{pre}}) = \max\left[v_{\text{suc}}\rho + \frac{1}{2}a_{\text{max,accel}}\rho^2 + \frac{\left(v_{\text{suc}} + \rho a_{\text{max,accel}}\right)^2}{2a_{\text{min,brake}}} - \frac{v_{\text{pre}}^2}{2a_{\text{max,brake}}}, 0\right]. \quad (4.12)$$

The *minimal safe lateral distance* [97, p. 11] is analogously given by

$$d_{\min}^{(\text{lat})}(v_{\text{lat,left}}, v_{\text{lat,right}}) = \mu + \max\left[s_{\text{right}}(v_{\text{lat,right}}) - s_{\text{left}}(v_{\text{lat,left}}), 0\right], \quad (4.13)$$

with

$$s_{\text{right/left}}(v_{\text{lat,right/left}}) = v_{\text{lat,right/left}}\rho \pm \frac{1}{2}a_{\text{lat,max,accel}}\rho^2 \pm \frac{\left(v_{\text{lat,right/left}} \pm a_{\text{lat,max,accel}}\rho\right)^2}{2a_{\text{lat,min,brake}}}. \quad (4.14)$$

For a quantitative determination of the safe distances multiple parameters must be defined. Their meaning is given in table 4.1. v_{suc} and v_{pre} are the velocities of the preceding and the succeeding vehicle in the longitudinal related case. Analogously $v_{\text{lat,left}}$ and $v_{\text{lat,right}}$ are the lateral velocities of the vehicle on the left and the right when considering their lateral relationship.

Table 4.1 Description of parameters of RSS. These parameters must be chosen in accordance to public agreements in order to make sure that the understanding of safety required by all stakeholders is consistent. Otherwise, RSS will not give comparable guarantees for safety. Based on table published by Jesenski et al. [134]; © 2020 IEEE.

Parameter	Description
ρ	response time (driver-dependent)
$a_{\text{max,accel}}$	maximally allowed longitudinal acceleration
$a_{\text{min,brake}}$	minimal longitudinal braking after ρ
$a_{\text{max,brake}}$	maximally allowed longitudinal braking
$a_{\text{lat,max,accel}}$	absolute of maximally allowed lateral acceleration
$a_{\text{lat,min,brake}}$	absolute of minimal lateral braking after ρ
μ	minimally allowed lateral distance

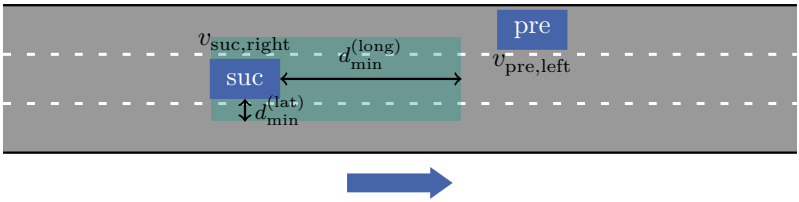


Figure 4.2 Longitudinal and lateral minimal safe distances demanded by RSS. If a pair of vehicles undercuts these minimal distances – that is pre gets into the green marked safety area around suc – both vehicles must respond accordingly. Failing to respond in accordance to RSS leads to the assignment of blame.

The safety area spanned by the lateral and longitudinal minimal safe distances is illustrated in fig. 4.2. If the vehicles get into this safety area both must react accordingly. That is, RSS defines a situation to be dangerous if both the lateral and longitudinal safe distance are undercut. The time when the second distance is undercut is called blame time t_{blame}. Depending on which of the both safe distances becomes undercut later, the vehicles must respond laterally or longitudinally [97, Definition 10]. If a longitudinal response becomes necessary, both vehicles can act mostly unconstrained during the response time ρ as long as they are accelerating with $a \in \left[-a_{\text{max,brake}}, a_{\text{max,accel}}\right]$. After the response time suc must decelerate with $a \in \left[-a_{\text{max,brake}}, -a_{\text{min,brake}}\right]$ [97, Definition 4]. If a lateral reaction is required both vehicles are allowed to act constrained only by

$a \in \left[-a_{\text{lat,max,accel}}, a_{\text{lat,max,accel}}\right]$ during the response time. Afterwards, both vehicles must accelerate laterally away from each other with at least $a_{\text{lat,min,brake}}$ and at most $a_{\text{lat,max,accel}}$[97, Definition 8]. Both types of reactions must endure until the relevant safe distance is again satisfied. If a vehicle does not comply with these demanded reactions, it gets the blame for a possible collision.

4.4.2 Physics-based metric based on RSS

Based on the definition of the minimal safe distances, a *physics-based RSS metric* can be constructed by

$$g_{\text{pair}}^{(\text{RSS Dist})}\left(p, t, \varphi_\xi\right) = \max_{\kappa \in \{\text{long,lat}\}} \frac{d_\kappa - d_{\text{min}}^{(\kappa)}}{d_{\text{min}}^{(\kappa)}}. \tag{4.15}$$

d_κ is the current lateral/longitudinal distance between the vehicles in pair p. Hence, $g_{\text{pair}}^{(\text{RSS Dist})}\left(p, t, \varphi_\xi\right) = -1$ corresponds to a 100% undercut of both minimal safe distances and thus represents a collision. A metric value of 0 is the border between an RSS violation and the satisfaction of the minimal safe distances. That means, all metric values larger 0 indicate safe scenes with satisfied safety areas (fig. 4.2). The range of the metric is $[-1, \infty)$. In order to handle infinite metric values during ordering, these values can be mapped to the largest finite values found in a dataset. Wishart et al. [123] have published a similar metric called *Minimum Safe Distance Factor*. The set of relevant vehicle pairs $\mathbf{A}_t^{(\varphi_\xi)}$ is given by the relevant vehicles in $\boldsymbol{\mathcal{S}}$ and its direct neighbors in the orange area as illustrated in fig. 4.3. In the next section, the physics-based metric in (4.15) will be used to define a behavior-based metric which considers blame.

4.4.3 Behavior-based metric

A behavior-based metric which solves the described limitations and requirements (section 3.3.3) can be expressed by

$$g_{\text{pair}}^{(\text{RSS Beh})}\left(p, t, \varphi_\xi\right) = \left[\alpha + \beta r_{\text{frac}}(p, t, \varphi_\xi) l(p, t, \varphi_\xi)\right] \cdot g_{\text{pair}}^{(\text{RSS Dist})}\left(p, t, \varphi_\xi\right). \tag{4.16}$$

α and β are parameters. Throughout the remainder it is set $\alpha = 1$ and $\beta = 9$. r_{frac} gives the fraction of time steps since $t_{\text{blame}}(p, t, \varphi_\xi) + \rho$ for

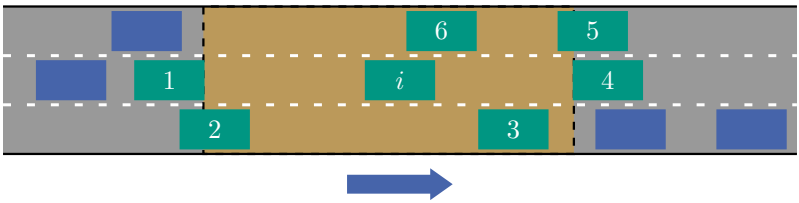


Figure 4.3 The vehicle pairs $(i,1),(i,2),\ldots$ in $\mathbf{A}_t^{(\varphi_\xi)}$ related to $i \in \boldsymbol{\mathcal{S}}$. The relevant vehicle pairs containing vehicle i are constrained to the green marked vehicles lying at least partly in the orange area which is defined by the direct predecessor and successor of i.

Table 4.2 Empirically selected parameter values for a full definition of the behavior-based metric given in (4.16) and (4.17).

Parameter	Ξ	k	η	Δ	α	β
Value	1	0.6	0.8	0.48 s	1	9

which the ego vehicle does not respond as demanded by RSS. In order to solve the second point of criticism in section 3.3.3, $l(p,t,\varphi_\xi)$ is assessing the length $t - t_{\text{blame}}$ of the dangerous scenario (how long have both distances been undercut?). For that, a weight of

$$l(p,t,\varphi_\xi) = \frac{\Xi}{1 + \exp\left(-k \cdot \Xi \cdot \frac{t - t_{\text{blame}}(p,t,\varphi_\xi) - \Delta}{\Delta t}\right) \cdot \left(\frac{\Xi}{\eta} - 1\right)} \tag{4.17}$$

is utilized. Table 4.2 gives the selected values for the metric parameters. The weighting l is depicted in fig. 4.4. If the scene is not critical ($g_{\text{pair}}^{(\text{RSS Dist})} \geq 0$) it follows that $r_{\text{frac}}(p,t,\varphi_\xi) = 0$ and $l(p,t,\varphi_\xi) = 0$. The sum $\left[\alpha + \beta r_{\text{frac}}(p,t,\varphi_\xi) l(p,t,\varphi_\xi)\right]$ therefore is necessary to make sure that the metric is still able to assess uncritical scenarios with $g_{\text{pair}}^{(\text{RSS Dist})}(p,t,\varphi_\xi) > 0$. In this case, $g_{\text{pair}}^{(\text{RSS Beh})}(p,t,\varphi_\xi) = \alpha \cdot g_{\text{pair}}^{(\text{RSS Dist})}(p,t,\varphi_\xi)$.

The third point of criticism in section 3.3.3 is also solved since it is possible to interpret different intervals in the range $g_{\text{pair}}^{(\text{RSS Beh})} \in [-\alpha - \beta, \infty)$ of the metric as follows: The most critical value of $-\alpha - \beta$ corresponds to a scenario with a colliding vehicle pair. Additionally, the ego vehicle is to blame in 100 % of critical time steps (however that does of course not imply that

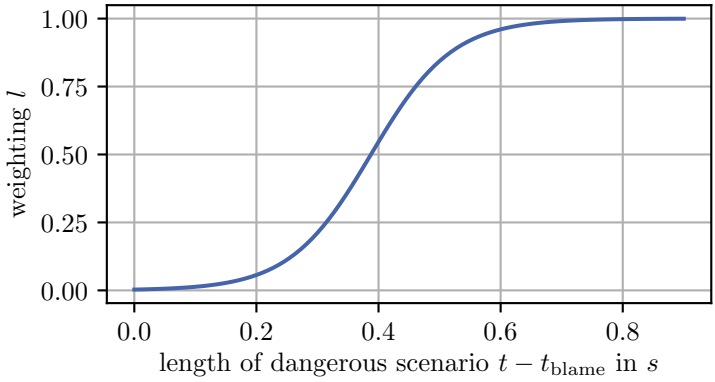


Figure 4.4 Illustration of $l(p, t, \xi)$ with parameters as given in table 4.2. The weighting l starts at 0 and increases to 1 after about 0.7 s.

the other vehicle in p is not to blame partially) and the critical situation endured longer than the characteristic time span encoded in $l(p, t, \varphi_\xi)$. For instance, with the parameters given in table 4.2, this characteristic time span needs to last about 0.8 s (see fig. 4.4). For the parameter range $g_{\text{pair}}^{(\text{RSS Beh})} \in [-\alpha - \beta, -\alpha)$ the ego vehicle must have reacted improperly at least once during the critical scenario after $t_{\text{blame}} + \rho$ $(r \cdot l > 0)$. For $(-\alpha, 0)$ this can be the case, but does not necessarily need to be so. As long as $g_{\text{pair}}^{(\text{RSS Beh})}(p, t, \xi) < 0$, both safety distances are undercut between the vehicles in p at time step t. For $g_{\text{pair}}^{(\text{RSS Beh})} \geq 0$ the scene is not critical and the safety area is satisfied at the selected time step.

Since (4.16) does not contain any (unnatural) comparisons it also fulfills the first point of criticism in section 3.3.3. Besides, the metric is continuous and therefore applicable to IS (section 5.3). The set of relevant vehicle pairs $\mathbf{A}_t^{(\varphi_\xi)}$ is chosen similar to $g_{\text{pair}}^{(\text{RSS Dist})}$.

5 Importance sampling

The simulation-based testing of ADFs requires the execution of rare event simulations. Section 1.2 already introduced the problem of efficiency when executing such simulations. As a possible solution, IS was discussed. However, as mentioned in section 3.3.2, in the current state of the art, importance sampling is still limited to scenarios of restricted complexity or a specific maneuver type. This chapter overcomes this challenge by focusing the importance sampling optimization onto the relevant parts of a scenario. In order to do so, section 5.1 shortly recapitulates the basic theory behind IS and section 5.2 proposes a new method which enables focusing on the relevant scenario parts during IS optimization. This should enable scalability of IS to highly complex scenarios. In section 5.3 the proposed method is then illustrated by implementing it on a simulation which is based on simple behavior models for time dynamics and which draws initial scenes from a dataset. Section 5.4 tackles the challenge to find the parts of a scenario which are relevant and therefore must be included into the respective IS optimization. This chapter is based on a publication by Jesenski et al. [134];

5.1 Theory of importance sampling

In this section, IS is introduced mathematically. For that, section 5.1.1 formally discusses the efficiency of naive rare event simulations. Section 5.1.2 mathematically describes IS and its capability to mitigate the efficiency problem. Finally, section 5.1.3 introduces the CE method and how it can be used to find the optimal IS distribution which is required to perform IS.

5.1.1 Efficiency problem of rare event simulations

The aim of simulations which apply statistical sampling (see section 3.2.1.1) is to calculate statistical evidence (see definition 3.2) e.g. about the safety

of an ADF. Mathematically, this can be expressed by the search for

$$P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}\right) = \mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right] = \int\limits_{\boldsymbol{\Omega}\times\boldsymbol{\Phi}} \mathfrak{I}\left(\varphi_\xi;\gamma\right) p_h\left(\varphi_h\right)\, \mathrm{d}\varphi_h, \quad (5.1)$$

with random variables $h = (\xi, \boldsymbol{\phi})$. The random variable ξ with $\varphi_\xi \in \boldsymbol{\Omega} = Val\left(\xi\right)$ are the physical properties of a fully parameterized simulation run h as already described in section 4.1. The random variables $\boldsymbol{\phi}$ with $\varphi_{\boldsymbol{\phi}} \in \boldsymbol{\Phi} = Val\left(\boldsymbol{\phi}\right)$ are the additional non-physical parameters which are necessary to fully describe a simulation. As will be discussed in section 5.3.1.1, that might be the parameters of the used behavior models. A fully assigned value $\varphi_h = \left(\varphi_\xi, \varphi_{\boldsymbol{\phi}}\right)$ is then an element of space $Val\left(h\right) = \boldsymbol{\Omega} \times \boldsymbol{\Phi}$. The distribution $p_h\left(\varphi_h\right)$ models the real occurrence rate of h. It must be extracted from measurements in reality. $P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}\right)$ then gives the probability that a critical set $\boldsymbol{\varepsilon}$ occurs. The critical event set $\boldsymbol{\varepsilon}$ whose probability of occurrence shall be calculated, is defined by

$$\boldsymbol{\varepsilon} = \left\{\varphi_\xi \in \boldsymbol{\Omega} : g_{\text{crit.}}\left(\varphi_\xi, \boldsymbol{\mathcal{S}}\right) \leq \gamma\right\}. \quad (5.2)$$

Thus, the indicator function defined by

$$\mathfrak{I}\left(\varphi_\xi;\gamma\right) = \begin{cases} 1 & \text{for } \varphi_\xi \in \boldsymbol{\varepsilon} \\ 0 & \text{otherwise} \end{cases} \quad (5.3)$$

allows selecting the relevant critical simulation runs. For that, a safety metric $g_{\text{crit.}}$ as introduced in chapter 4 becomes necessary.

By running MCS, the required expectation value $\mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]$ can be approximated by the random variable

$$Y_N = \frac{1}{N}\sum_{i=1}^{N} \mathfrak{I}\left(\xi_i;\gamma\right); \qquad h_i = \left(\xi_i, \boldsymbol{\phi}_i\right) \sim p_h, \quad (5.4)$$

since

$$\mathbb{E}_{p_{Y_N}\left(Y_N\right)}\left[Y_N\right] = \mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]. \quad (5.5)$$

The random variables $\{\xi_1, ..., \xi_N\}$ are obviously independently and identically (i.i.d) distributed. The parameter N gives the number of performed

simulation runs drawn from p_h. The central limit theorem [96, section 1.2.2] can be used to show that for $N \to \infty$ the distribution $p_{Y_N}\left(Y_N\right)$ of Y_N is determined by a Gaussian distribution with relative standard deviation[1] [12] of

$$\sigma_{\text{dev,rel}}\left[Y_N\right] = \frac{\sqrt{\text{Var}_{p_{Y_N}(Y_N)}\left[Y_N\right]}}{\mathbb{E}_{p_{Y_N}(Y_N)}\left[Y_N\right]} = \sqrt{\frac{1 - \mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]}{\mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right] \cdot N}}. \quad (5.6)$$

Since the denominator in (5.6) is proportional to the probability of occurrence $P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}\right) = \mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]$, a rarely occurring event $\boldsymbol{\varepsilon}$ will cause large relative standard deviations in the results obtained by simulations. Hence, in order to get stable results, the number of samples N must be chosen very large. This however, will cost a lot of resources and simulation time. In the next section, IS is discussed in detail. It is shown, that IS is able to strongly reduce the relative standard deviation and thereby the number of samples required to obtain statistically valid results can be minimized.

5.1.2 Importance sampling

IS can be applied in order to reduce the size of the relative standard deviation in (5.6). This can be achieved by reducing the sampling of uncritical and irrelevant simulation runs and thus by focusing on critical scenarios. For that, the original distribution $p_h\left(\varphi_h\right)$ is skewed to the more critical distribution $p_h^*\left(\varphi_h\right)$. By using

$$\mathcal{K}_*\left(\varphi_h\right) = \frac{p_h\left(\varphi_h\right)}{p_h^*\left(\varphi_h\right)} \quad (5.7)$$

in

$$\begin{aligned} P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}\right) &= \mathbb{E}_{p_h^*(h)}\left[\mathfrak{I}\left(\xi;\gamma\right) \cdot \mathcal{K}_*\left(h\right)\right] \\ &= \int\limits_{\boldsymbol{\Omega}\times\boldsymbol{\Phi}} \mathfrak{I}\left(\varphi_\xi;\gamma\right) \mathcal{K}_*\left(\varphi_h\right) p_h^*\left(\varphi_h\right)\, \mathrm{d}\varphi_h, \end{aligned} \quad (5.8)$$

[1] The relative standard deviation is defined to be the standard deviation of a random variable divided by the random variable's expectation value.

it is still possible to get statistical statements about the original distribution p_h. (5.8) can be approximated by the random variable

$$Y_N^* = \frac{1}{N} \sum_{i=1}^{N} \mathfrak{I}\left(\xi_i;\gamma\right) \mathcal{K}_*\left(h_i\right); \qquad h_i = \left(\xi_i, \boldsymbol{\phi}_i\right) \sim p_h^*. \tag{5.9}$$

The original results are reproduced since

$$\begin{aligned} \mathbb{E}_{p_{Y_N^*}\left(Y_N^*\right)}\left[Y_N^*\right] &= \mathbb{E}_{p_h^*(h)}\left[\mathfrak{I}\left(\xi;\gamma\right) \cdot \mathcal{K}_*\left(h\right)\right] \\ &= \mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]. \end{aligned} \tag{5.10}$$

The benefit is, that for $N \to \infty$, $p_{Y_N^*}\left(\varphi_{Y_N^*}\right)$ also has a Gaussian shape with a relative standard deviation [12] of

$$\begin{aligned} \sigma_{\mathrm{dev,rel}}\left[Y_N^*\right] &= \frac{\sqrt{\mathrm{Var}_{p_{Y_N^*}}\left[Y_N^*\right]}}{\mathbb{E}_{p_{Y_N^*}}\left[Y_N^*\right]} \\ &= \sqrt{\frac{1}{N} \cdot \left(\frac{\lambda}{\left(\mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]\right)^2} - 1 \right)}, \end{aligned} \tag{5.11}$$

with $\lambda = \mathbb{E}_{p_h^*(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)^2 \cdot \mathcal{K}_*\left(h\right)^2\right]$. If the optimal IS distribution

$$p_{h,\mathrm{opt}}^*\left(\varphi_h\right) = \frac{p_h\left(\varphi_h\right)}{\mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]} \cdot \mathfrak{I}\left(\varphi_\xi;\gamma\right) \tag{5.12}$$

is used, then $\lambda = \left(\mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]\right)^2$ and the relative standard deviation vanishes and perfect results would be obtained from one evaluation of Y_N^*. Since it is not possible to find this optimal IS distribution ($\mathbb{E}_{p_h(h)}\left[\mathfrak{I}\left(\xi;\gamma\right)\right]$ is the result of simulations), the challenge is to find a good approximation of the optimal IS distribution which minimizes $\sigma_{\mathrm{dev,rel}}\left[Y_N^*\right]$. One of the most often used methods to find such an approximation is the cross entropy (CE) method.

5.1.3 Cross entropy method

A popular approach to approximate the optimal IS distribution (5.12) is the cross entropy method (CE) [15, 89]. In general, CE tries to find the optimal member $q\left(\varphi_h \middle| \mathfrak{v}^*\right)$ within a family of functions $q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right)$ whose "deviation", as defined by the KBL, to (5.12) is minimal. It can be shown [15], that the optimal parameter values $\mathfrak{v}^*$ can be found by optimizing

$$\mathfrak{v}^* = \arg\max_{\varphi_{\mathfrak{v}}} D\left(\varphi_{\mathfrak{v}}, \boldsymbol{\omega}, \gamma\right) \tag{5.13}$$

with

$$D\left(\varphi_{\mathfrak{v}}, \boldsymbol{\omega}, \gamma\right) = \mathbb{E}_{q(h|\mathfrak{v}=\boldsymbol{\omega})}\left[\mathfrak{I}\left(\xi;\gamma\right)\mathcal{K}\left(h;\boldsymbol{\omega}\right)\ln\left(q\left(h|\varphi_{\mathfrak{v}}\right)\right)\right]. \tag{5.14}$$

$$\approx \frac{1}{\mathfrak{N}}\sum_{i=1}^{\mathfrak{N}} \mathfrak{I}\left(\varphi_{\xi_i};\gamma\right)\mathcal{K}\left(\varphi_{h_i};\boldsymbol{\omega}\right)\ln\left(q\left(\varphi_{h_i}\middle|\varphi_{\mathfrak{v}}\right)\right) \tag{5.15}$$

with

$$h_i = \left(\xi_i, \boldsymbol{\phi}_i\right) \sim q\left(h|\mathfrak{v}=\boldsymbol{\omega}\right). \tag{5.16}$$

It is

$$\mathcal{K}\left(\varphi_h;\boldsymbol{\omega}\right) = \frac{p_h\left(\varphi_h\right)}{q\left(\varphi_h\middle|\mathfrak{v}=\boldsymbol{\omega}\right)}. \tag{5.17}$$

In principle, an arbitrary member $q\left(\varphi_h\middle|\boldsymbol{\omega}\right)$ could be chosen for the optimization. However, for numerical stability a multilevel approach as defined in algorithm 1 is applied. In this approach, the optimization in (5.13) is executed iteratively. For each iteration i, the value γ_i which defines an event set $\boldsymbol{\varepsilon}_i$ is further reduced until the required value γ is finally reached. This iterative reduction of γ allows creating a higher proportion of relevant samples with $\mathfrak{I}\left(\xi;\gamma\right) = 1$ for each iteration. This is necessary for a stable optimization. Otherwise, not enough data points would be available to get a good approximation of the expectation value in (5.14). The inputs to algorithm 1 are the quantile η, the number of samples per iteration $\mathfrak{N}$ which are used to determine the expectation value, the maximum number of iterations $k_{\max}$, the unbiased density $p_h\left(\varphi_h\right)$, a safety metric $g_{\text{crit.}}$ and the required level of criticality γ which defines $\boldsymbol{\varepsilon}$. The algorithm outputs $\mathfrak{v}^* = \mathfrak{v}_k$. The density $q\left(\varphi_h\middle|\mathfrak{v}^*\right) \approx p^*_{h,\text{opt}}\left(\varphi_h\right)$ can then be used to calculate Y^*_N in (5.9).

Algorithm 1 Cross entropy algorithm; algorithm 2.1 in [15]

procedure CROSSENTROPY($\mathfrak{N}, \eta, k_{\max}, p_h, g_{\text{crit.}}, \gamma$)
  $p^{(0)} = p_h$; $k = 0$
  while $k < k_{\max}$ **do**
    sample $\boldsymbol{h}_{\text{sample},k} \leftarrow \left\{\varphi_{h_1}, \ldots, \varphi_{h_{\mathfrak{N}}}\right\} \sim p^{(k)}$
    $\gamma_k \leftarrow \eta$-quantile of $\left\{g_{\text{crit.}}\left(\varphi_\xi, \boldsymbol{\mathcal{S}}\right) : (\varphi_\xi, \varphi_{\boldsymbol{\phi}}) \in \boldsymbol{h}_{\text{sample},k}\right\}$
    if $\gamma_k < \gamma$ **then**
      $\gamma_k \leftarrow \gamma$
    end if
    $\mathfrak{v}_k \leftarrow \arg\max_{\varphi_{\mathfrak{v}}} D(\varphi_{\mathfrak{v}}; \mathfrak{v}_{k-1}, \gamma_k)$ use $\boldsymbol{h}_{\text{sample},k}$
    if $\gamma_k \leq \gamma$ **then**
      break
    end if
    $k \leftarrow k + 1$
    $p^{(k)} = q\left(\varphi_h \middle| \mathfrak{v}_{k-1}\right)$
  end while
  return $\mathfrak{v}_k$
end procedure

5.2 Extension to high dimensional ADF-simulations

This section recalls the scaling problem of CE and proposes solutions. For that, section 5.2.1 proposes the concept of *causality groups* which allows focusing the CE optimization on the "relevant parts" of a simulation run. Section 5.2.2 adds some adaptions to CE which improve the numerical stability of the optimization procedure.

5.2.1 Causality groups

In general, the described CE method scales poorly when applied to high-dimensional, complex simulations. Since simulations which evaluate the safety of an automated vehicle and its surrounding have a high number of parameters this scaling challenge heavily affects them. In order to address

this challenge, this section proposes two adjustments. Firstly, only *relevant*[2] parts of the total probability distribution of a simulation run shall be optimized which allows reducing the number of dimensions which have to be considered when performing CE. Secondly, the separation of these *relevant* parts into *causality groups* is proposed.

Simulations of automated vehicle behavior typically are determined by distributions of type

$$p_h\left(\varphi_h\right) = p_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}}\right) \cdot p_{\text{beh.}}\left(\varphi_{\boldsymbol{x}_1}, ..., \varphi_{\boldsymbol{x}_{n_{\varphi_\xi}}} \middle| \varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}_{\text{beh.}}}\right). \quad (5.18)$$

Remember, that it is $\varphi_\xi = \left(\varphi_{\boldsymbol{x}_0}, ..., \varphi_{\boldsymbol{x}_{n_{\varphi_\xi}}}\right) \in \boldsymbol{\Omega}$. The random variable $\boldsymbol{x}_t$ gives the scene at time t of simulation run h, as already introduced in section 4.1. The factorization into p_0 and $p_{\text{beh.}}$ is based on the assumption that the initial scene and time dynamics are described by different models: Scene models and behavior models. The parameters $\boldsymbol{\phi}_{\text{beh.}} \subseteq \boldsymbol{\phi}$ represent the non-physical parameters which have an influence on the dynamics of a simulation run.

The parametric family of distributions used for CE optimization can of course be chosen arbitrarily. In analogy to (5.18) however, it makes sense to choose

$$q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right) = q_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}} \middle| \varphi_{\mathfrak{v}}\right) \cdot p_{\text{beh.}}\left(\varphi_{\boldsymbol{x}_1}, ..., \varphi_{\boldsymbol{x}_{n_{\varphi_\xi}}} \middle| \varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}_{\text{beh.}}}\right), \quad (5.19)$$

in order to enable good approximations of the optimal IS distribution. As already mentioned, $\mathfrak{v}$ may have a very high dimension. Hence, the resulting high-dimensional optimization of $\mathfrak{v}$ in (5.13) is infeasible or at least highly complicated since the "volume" of the search space increases exponentially with $|\mathfrak{v}|$.

As an example for the possible size of $\mathfrak{v}$'s dimension, an estimate of the dimension $|\mathfrak{v}|$ was given for a straight highway section in section 1.3. It was assumed that ten vehicles governed by the intelligent driver model are included in the surrounding of an automated ego vehicle. It was further supposed that the skewed distribution's parameters only describe the

[2] A parameter is *relevant* in this context if a value change of this parameter has significant influence on the sampling of simulation runs with critical safety metric values $g_{\text{crit.}}$. A quantitative method to find such parameters will be discussed in section 5.4.

behavior parameters of these surrounding vehicles by using truncated Gaussian mixture distributions with three summands per behavior parameter. Consequently, it was shown that $|\mathfrak{v}|$ increases already to 630 dimensions. Highway scenarios as for example included in the highD dataset can contain 30+ vehicles. That means, the number of parameters could become much higher, additionally when considering parameters which are not related to the behavior models.

In order to reduce $|\mathfrak{v}|$'s size, the optimization now shall be limited to the parts of h which are *relevant* for safety. A separation into relevant and irrelevant parts should be possible for most ADF simulations since for example the behavior parameters of a vehicle 200 m behind the ego vehicle are less important than the behavior parameters of its direct predecessor. After having found such a separation, it is useful to reformulate

$$\begin{aligned} p_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}}\right) =& p_{0,\text{irr.}}\left(\varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right) \\ & \cdot p_{0,\text{rel.}}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.}}}, \varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right). \end{aligned} \tag{5.20}$$

It is $\boldsymbol{x}_0 = \boldsymbol{x}_{0,\text{irr.}} \cup \boldsymbol{x}_{0,\text{rel.}}$ and $\boldsymbol{\phi} = \boldsymbol{\phi}_{\text{irr.}} \cup \boldsymbol{\phi}_{\text{rel.}}$. The distribution $p_{0,\text{irr.}}$ describes the random variables $\boldsymbol{x}_{0,\text{irr.}}, \boldsymbol{\phi}_{\text{irr.}}$ which are approximately *irrelevant* for the safety of the ego vehicle and $p_{0,\text{rel.}}$ represents the distribution of parameters $\boldsymbol{x}_{0,\text{rel.}}, \boldsymbol{\phi}_{\text{rel.}}$ which are *relevant* for safety. This separation between relevant and irrelevant parameters can then be exploited by analogously defining

$$\begin{aligned} q_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}} \middle| \varphi_{\mathfrak{v}}\right) =& p_{0,\text{irr.}}\left(\varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right) \\ & \cdot q_{0,\text{rel.}}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.}}}, \varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}, \varphi_{\mathfrak{v}}\right). \end{aligned} \tag{5.21}$$

Note that $p_{0,\text{irr.}}$ is kept to be the same as in the unskewed distribution. Since it only contains irrelevant parameters it does not have to be optimized. This alone already might strongly reduce the total number of parameters $|\mathfrak{v}|$ since $\mathfrak{v}$ would become larger if the number of relevant parameters which must be optimized increased[3]. The reason behind that is, that for models $q_{0,\text{rel.}}$ to be expressive, the number of model parameters $\mathfrak{v}$ increases with the distribution's dimension and complexity which is given by the size of $\boldsymbol{x}_{0,\text{rel.}}, \boldsymbol{\phi}_{\text{rel.}}$.

[3] The standard case, which assumes all parameters in $\boldsymbol{x}_0, \boldsymbol{\phi}_h$ are relevant, is of course the worst case.

As already mentioned, the second proposition to tackle the dimension problem is the introduction of *causality groups*. Since simulations are often built in a modular fashion (e.g. by defining individual (behavior) models and parameterizations separately for each vehicle in a simulation run), it is intuitive to assume that there are subsets of relevant parameters $c \in \mathfrak{C}$ which are independent of each other. Such a subset c shall be called *causality group* and it gives the subset of relevant parameters which are dependent on each other, but independent from parameters of other causality groups. The parameters of a causality group therefore must be handled in a joint distribution. By definition $\mathfrak{C}$ shall contain all causality groups. Consequently, it is possible to write

$$q_{0,\text{rel.}}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.}}}, \varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}, \varphi_{\mathfrak{v}}\right) = \prod_{c \in \mathfrak{C}} q_{0,\text{rel.},c}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.},c}}, \varphi_{\boldsymbol{\phi}_{\text{rel.},c}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}, \varphi_{\mathfrak{v}^{(c)}}\right). \quad (5.22)$$

It is $\boldsymbol{x}_{0,\text{rel.}} = \bigcup_{c \in \mathfrak{C}} \boldsymbol{x}_{0,\text{rel.},c}$ and $\boldsymbol{\phi}_{\text{rel.}} = \bigcup_{c \in \mathfrak{C}} \boldsymbol{\phi}_{\text{rel.},c}$.

It can be shown that this enables reducing and splitting the optimization in (5.13) to

$$\mathfrak{v}^{(c,*)} = \arg\max_{\varphi_{\mathfrak{v}^{(c)}}} D_c\left(\varphi_{\mathfrak{v}^{(c)}}, \boldsymbol{\omega}, \gamma\right) \forall c \in \mathfrak{C} \quad (5.23)$$

with $\mathfrak{v}^* = \left(\mathfrak{v}^{(c_1,*)}, ..., \mathfrak{v}^{(c_{|\mathfrak{C}|},*)}\right)$. Besides, it is

$$D_c\left(\varphi_{\mathfrak{v}^{(c)}}, \boldsymbol{\omega}, \gamma\right) = \mathbb{E}_{q(h|\mathfrak{v}=\boldsymbol{\omega})}\left[\mathfrak{I}\left(\xi;\gamma\right) \mathcal{K}_{\text{rel.}}\left(h;\boldsymbol{\omega}\right) \ln\left(q_{0,\text{rel.},c}\left(...|...\right)\right)\right], \quad (5.24)$$

and

$$\mathcal{K}_{\text{rel.}}\left(\varphi_h;\boldsymbol{\omega}\right) = \frac{p_{0,\text{rel.}}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.}}}, \varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right)}{q_{0,\text{rel.}}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.}}}, \varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}, \mathfrak{v} = \boldsymbol{\omega}\right)}. \quad (5.25)$$

Consequently, one optimization of dimension $|\mathfrak{v}|$ could be split up to $|\mathfrak{C}|$ optimizations with dimensions $\left|\mathfrak{v}^{(c)}\right|$. The split up into causality groups depends on the structure of the simulation and on the assumptions regarding

the correlation of relevant parameters with regard to the critical scenes. The best gains can be achieved when splitting the simulation into very small causality groups which for example might include only the relevant parameters of one vehicle or even only one relevant parameter at all. Then the dimensions of the single optimizations $\left|\mathfrak{v}^{(c)}\right|$ become as small as possible. When doing so, the factorization of the total distribution cannot in principle falsify the results of (5.9) when the number of samples N is large enough. However, a poor approximation of (5.12) might strongly increase the relative standard deviation in (5.11). This might cause severe underestimations of $P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}\right)$ when the number of samples is too low to find critical simulation runs in underrepresented parameter areas.

Section 5.3.2 will exemplarily implement the described separation procedure and show the resulting reduction in the dimension of the required optimizations.

One might wonder why one should not try to delete or at least reduce the number of irrelevant parameters in the models in order to reduce the dimensionality of the optimization (by using different/smaller models). Such a parameter reduction however would require the construction of models p_0 and $p_{\text{beh.}}$ which do not strive to represent reality as exact and generic as possible, but only the parts of reality which are critical for a specific ADF (e.g. only the vehicles/ the vehicle configurations which affect the ADF e.g. in a maneuver-based way). This however would require good knowledge of the critical behavior of the ADF before creating the models and doing the simulations. If no complete knowledge about the weaknesses of the ADF was available, the resulting models might not be representative and neglect some important critical aspects of reality which then will be overlooked and cannot be optimized. When applying IS this can lead to severe underestimations of the collision rate since no good optimal IS distribution can be found (for this reason also methods as described in section 5.4 become necessary). However, if complete knowledge about the critical behavior was available, no simulations at all would be necessary since the system was already understood perfectly beforehand. In addition, since the limitation on ADF-specific critical parts of reality would prevent using generic models and the weaknesses of an ADF might change throughout its development, adapted simulation models might be required for each development version of the ADF. The same also holds true when

calculating the occurrence rates of differently defined critical event sets[4] $\boldsymbol{\varepsilon}$. Additionally, completely new behavior models would be needed since expressive and generic behavior models which try to model reality as good as possible need a certain set of parameters (separately for each vehicle). Just leaving out a parameter here is not possible. Hence, in the case of an intended behavioral parameter reduction an extra behavior model would be needed for each surrounding vehicle at a certain position in relation to the ADF vehicle. The reason for that is, that different behavior is relevant for different vehicles, e.g. a vehicle on the lane to the left of the ego vehicle is critical if it fastly changes lanes to the right. A vehicle, which precedes the ego vehicle, would be dangerous if it decelerated very strongly. Summarized, parameter reduction prevents using generic models and requires the usage of maneuver-based models. The described parameter explosion is however inevitable when using generic models.

5.2.2 Adaptions to the optimization procedure

Despite the dimension of the necessary optimization, there are some other hurdles which prevent the efficient application of IS to high-dimensional simulation frameworks. In the following, solutions are proposed. They will be applied in the experiments discussed in section 5.3.4.

5.2.2.1 Numerical instability of weighting

One hurdle is the numerical instability of the relation $\mathcal{K}_{\text{rel.}}\left(\varphi_h;\boldsymbol{\omega}\right)$. When sampling, the samples of $\mathcal{K}_{\text{rel.}}\left(\varphi_h;\boldsymbol{\omega}\right)$ might take largely different values, especially when the distributions $p_{0,\text{rel.}}$ and $q_{0,\text{rel.}}$ contain a large number of relevant parameters in $\boldsymbol{x}_{0,\text{rel.}}, \boldsymbol{\phi}_{\text{rel.}}$. That can however result in only very few simulation runs with large $\mathcal{K}_{\text{rel.}}\left(\varphi_h;\boldsymbol{\omega}\right)$ dominating the calculation of the expectation value in (5.24) or (5.14). Consequently, these outliers should be filtered. For that, the starting point is the set of critical simulation runs

$$\boldsymbol{h}_{\text{sample,crit.,k}} = \left\{\varphi_h \in \boldsymbol{h}_{\text{sample,k}} : \mathfrak{I}\left(\varphi_\xi;\gamma_k\right) = 1\right\}, \tag{5.26}$$

with the set $\boldsymbol{h}_{\text{sample,k}}$ used for the optimization in iteration k (see algorithm 1). The simulation runs with $\mathfrak{I}\left(\varphi_\xi;\gamma\right) = 0$ can be neglected since these values do not influence the expectation value when changing the value

[4] e.g. for different accident types

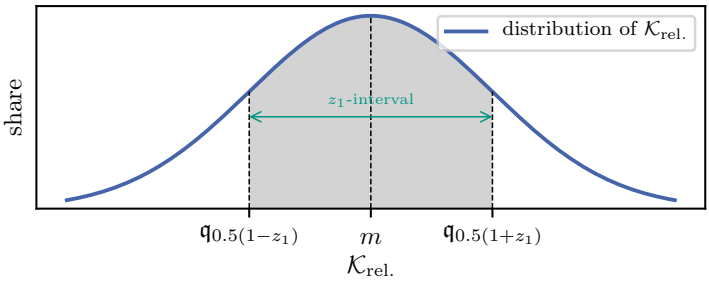


Figure 5.1 Illustration of the z_1-interval around the median m. Note, that $\mathcal{K}_{\text{rel.}} \in \mathfrak{K}$ can be distributed arbitrarily. For illustrative purposes only, a Gaussian distribution was chosen here.

of $\mathfrak{v}$ during the optimization procedure. The respective weightings define the set

$$\mathfrak{K} = \left\{ \mathcal{K}_{\text{rel.}} \left(\varphi_h ; \boldsymbol{\omega} \right) : \varphi_h \in \boldsymbol{h}_{\text{sample,crit.,k}} \right\} . \tag{5.27}$$

This set's median (0.5-quantile) shall be denoted by

$$m = \mathfrak{q}_{0.5} \left(\mathfrak{K} \right) . \tag{5.28}$$

First of all, the values located above z_0 times the upper limit value of the z_1-quantile around m are filtered[5]. This upper limit is equivalent to the upper limit of $\mathfrak{q}_{0.5(1+z_1)} \left(\mathfrak{K} \right)$. The filtering can mathematically be expressed by the resulting set

$$\mathfrak{K}_{\text{filter}} = \left\{ \mathcal{K} \in \mathfrak{K} : \mathcal{K} - m > z_0 \cdot \left(\mathfrak{q}_{0.5(1+z_1)} \left(\mathfrak{K} \right) - m \right) \right\} . \tag{5.29}$$

The used z_1-interval is sketched in fig. 5.1. Finally, all values in $\mathfrak{K}_{\text{filter}}$ whose distance to the next smaller neighbor is larger than

$$d_{\max} = z_2 \cdot \left(\mathfrak{q}_{0.999} \left(\mathfrak{K} \right) - m \right) \tag{5.30}$$

shall be selected. Therefore the next lower neighbour of $\mathcal{K}$ in a set $\boldsymbol{A}$ of weightings is selected by

$$f_{\text{lower}} \left(\mathcal{K}, \boldsymbol{A} \right) = \max \left(\{ a \in \boldsymbol{A} : a < \mathcal{K} \} \right) . \tag{5.31}$$

[5] Note, that z_0 and z_1 are parameters which can be empirically chosen.

The smallest value in $\mathfrak{K}_{\text{filter}}$ which has a distance to the next smaller neighbor larger then $d_{\max}$ shall then be extracted by

$$\mathcal{K}_{\text{limit}} = \min\left(\left\{\mathcal{K} \in \mathfrak{K}_{\text{filter}} : \mathcal{K} - f_{\text{lower}}\left(\mathcal{K}, \mathfrak{K}\right) > d_{\max}\right\}\right). \tag{5.32}$$

This smallest value in $\mathfrak{K}_{\text{filter}}$ is then used as cut-off value in

$$\boldsymbol{h}_{\text{sample,filtered,k}} = \left\{\varphi_h \in \boldsymbol{h}_{\text{sample,crit.,k}} : \mathcal{K}_{\text{rel.}}\left(\varphi_h; \boldsymbol{\omega}\right) < \mathcal{K}_{\text{limit}}\right\}. \tag{5.33}$$

The set $\boldsymbol{h}_{\text{sample,filtered,k}}$ instead of $\boldsymbol{h}_{\text{sample,k}}$ is now used to approximate the expectation values (5.24) needed for the optimization[6]. In summary, outliers which exceed a certain distance to the next lower value are thus neglected. For the calculation of the respective sets, quantiles, the median and distances in between were used instead of expectation values and standard deviations. The reason is, that quantiles are largely invariant with regard to outliers. In contrast, outliers can strongly influence expectation values. This might directly prevent these outliers from being excluded when using expectation values. In the remainder of this work, the filtering procedure is parameterized empirically by

$$z_0 = 2, \tag{5.34}$$
$$z_1 = 0.96, \tag{5.35}$$
$$z_2 = 0.25. \tag{5.36}$$

5.2.2.2 Exploration vs exploitation

The optimization procedure in algorithm 1 iteratively reduces the deviation of the sampling distribution $q\left(\varphi_h|\mathfrak{v}_{k-1}\right)$ to the optimal importance sampling distribution by iteratively reducing the accepted criticality limit γ_k. A disadvantage of this iterative process is, that critical areas in parameter space which possibly have been left completely undiscovered in iteration $k-1$ probably will not be found in iteration k. The reason is that during iteration $k-1$, the distribution $q\left(\varphi_h|\mathfrak{v}_{k-1}\right)$ will be optimized to contain high probabilities only for parameter values φ_h which have been discovered to be

[6] The rest of the samples with $\mathfrak{I}\left(\varphi_\xi; \gamma\right) = 0$ can be ignored since they only add 0 to the sum in (5.15). Hence, they only contribute to the expectation value by changing the factor $\frac{1}{\mathfrak{N}}$. When applying $\arg\max_{\mathfrak{v}^{(c)}}$..., this factor however can be neglected and therefore it is sufficient to just use $\boldsymbol{h}_{\text{sample,filtered,k}}$.

critical ($g_{\text{crit.}} \leq \gamma_{k-1}$) during iteration $k-1$. The other areas' occurrence probability will be reduced. Since $q\left(\varphi_h|\mathfrak{v}_{k-1}\right)$ is used to draw samples for the optimization in iteration k, these already emphasized areas from iteration $k-1$ will mainly determine the kth optimization. Therefore, the even more critical parameter areas with $g_{\text{crit.}} \leq \gamma_k$ which are found during iteration k tend to be located only within the already emphasized parameter ranges. If there were some critical areas which were not found in iteration $k-1$, it would be very unlikely to find them now by chance. That means, not all relevant critical parameter areas might be found during the optimization procedure if some critical parameter areas were not identified in early iterations. Eventually that means, that the amount of exploration during an iteration is limited.

Additionally, if the numerical instability of $\mathcal{K}_{\text{rel.}}$ introduced distortions into $q\left(\varphi_h|\mathfrak{v}_{k-1}\right)$ when optimizing in iteration $k-1$, this directly influences the samples in iteration k and thereby affects the construction of $q\left(\varphi_h|\mathfrak{v}_k\right)$.

Due to the use of importance sampling within (5.24), it is in principle possible to sample from arbitrary distributions without distorting the expectation value. This fact is now used to propose an adaption of the sampling in order to mitigate the mentioned challenges. For that, the sampling distribution $q\left(\varphi_h|\mathfrak{v}_{k-1}\right)$ for iteration k is changed by adding N_{add} distributions $q_{\text{add},i}$ to $q_{0,\text{rel.},c}$. The resulting distribution is given by

$$\begin{aligned}\tilde{q}_{0,\text{rel.},c}&\left(\varphi_{\boldsymbol{x}_{0,\text{rel.},c}}, \varphi_{\boldsymbol{\phi}_{\text{rel.},c}} \middle| \varphi_{\boldsymbol{x}_{0,\text{irr.}}}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}, \mathfrak{v}_{k-1}^{(c)}\right) \\ &= \sum_{i=1}^{N_{\text{add}}} \beta_i \cdot q_{\text{add},i}\left(...\right) + \beta_0 \cdot q_{0,\text{rel.},c}\left(...\right)\end{aligned} \tag{5.37}$$

with $0 \leq \beta_i \leq 1$ and $\sum_{i=0}^{N_{\text{add}}} \beta_i = 1$. In the remainder, this work proposes the use of $N_{\text{add}} = 2$. Firstly, the uniform distribution

$$q_{\text{add},1}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.},c}}, \varphi_{\boldsymbol{\phi}_{\text{rel.},c}}\right) = \mathcal{U}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.},c}}, \varphi_{\boldsymbol{\phi}_{\text{rel.},c}}\right) \tag{5.38}$$

is added with weight β_1. The application of the uniform distribution enhances the algorithm's ability to explore new parameter ranges which have not previously been known to contain critical simulation runs.

Secondly, a histogram *hist* (in a strict mathematical sense it will later be a categorical distribution obtained after discretizing the variables) over the critical simulation runs of the last iteration $k-1$

$$q_{\text{add},2}\left(\varphi_{\boldsymbol{x}_{0,\text{rel.},c}}, \varphi_{\boldsymbol{\phi}_{\text{rel.},c}}\right) = hist\left(\varphi_{\boldsymbol{x}_{0,\text{rel.},c}}, \varphi_{\boldsymbol{\phi}_{\text{rel.},c}} \middle| \boldsymbol{h}_{\text{sample,crit.,k}-1}\right) \tag{5.39}$$

is added with weight β_2. The addition of $q_{\text{add},2}$ improves the overall stability and convergence rate of the algorithm since it reduces the influence of the unstable factor $\mathcal{K}_{\text{rel.}}$. In contrast to $q_{0,\text{rel.},c}$, $q_{\text{add},2}$ does not care about the weightings of the samples. It purely steers the distribution to critical parameter regions which have been sampled most frequently in the last iteration. Hence, β_0, β_1 and β_2 can be used to weight the algorithm's amounts of exploration and exploitation[7].

5.2.2.3 Premature convergence

Another challenge of the iterative optimization procedure is that γ_k might undercut the final value γ before $\mathfrak{v}_{k-1}$ has converged and thus before the deviation of $q\left(\varphi_h|\mathfrak{v}_{k-1}\right)$ to the optimal importance sampling distribution has fallen below an acceptable level. New critical parameter areas might be found after reaching γ since some parameter ranges only become interesting after some correlated parameters have been optimized into a certain range. Therefore, this work proposes the execution of an additional number k_{add} of iterations which take places after reaching $\gamma_k \leq \gamma$ at iteration $k = k_{\text{reach}}$. For the iterations $b \in \{k_{\text{reach}}, ..., k_{\text{reach}} + k_{\text{add}}\}$ the value of γ_b is, in accordance with algorithm 1, set to γ. Within the remainder of this work

$$k_{\text{add}} = 2 \tag{5.40}$$

is utilized. During the additional iterations, the number of performed simulations used to calculate the respective expectation value is given by $\mathfrak{N}_{\text{add}}$. Within this work,

$$\mathfrak{N}_{\text{add}} = 20\,000 \tag{5.41}$$

is used.

5.3 Implementation and illustration of CE on simple behavior models

In this section, the mentioned improvements enhancing the scalability of CE shall be illustrated. This is accomplished by applying them to determine the rare collision rate in simulations of an ADF-equipped ego vehicle in complex

[7] exploration: β_1; exploitation: β_0, β_2

highway scenes. The initial scenes of the simulation runs are directly taken from the highD dataset (for the dataset, see appendix B.2, for the sampling of initial scenes see section 5.3.1.1). This makes sure that realistic scenes which are not limited to certain scenario classes or complexities are used. For the desired proof of concept, the vehicle dynamics are governed by simple behavior models as described in section 5.3.1. The concrete causality groups used to enable the scalability are introduced within section 5.3.2. When taking initial scenes from a dataset it becomes impossible to optimize them. Therefore, a *relaxation phase* (section 5.3.3) becomes necessary. The results of the optimization procedure and the ability to accelerate the generation of statistically stable results is then evaluated on two metrics (section 5.3.4): A purely physics-based one (section 5.3.4.1) and a behavior-based one (section 5.3.4.2).

5.3.1 Model

5.3.1.1 Initial scenes

At the beginning of each simulation run an initial scene must be constructed. These initial scenes, which represent the physical vehicle configuration $\boldsymbol{x}_0$ at the beginning of a simulation run h, are constructed by drawing arbitrary samples from the highD dataset (appendix B.2). That implies, $\boldsymbol{x}_0 \perp\!\!\!\perp \boldsymbol{\phi}$. For the sake of simplicity, in this first illustration of the concept, only non-physical parameters $\boldsymbol{\phi}_{\text{rel.}} \subseteq \boldsymbol{\phi}$ shall be optimized by CE. Hence, this implicitly assumes that the initial scene configuration is totally irrelevant for the safety of the ADF-equipped vehicle under test[8]. Hence, it is $\boldsymbol{x}_{0,\text{rel.}} = \{\}, \boldsymbol{x}_{0,\text{irr.}} = \boldsymbol{x}_0$. As a result of these assumptions, (5.20) can be specified

[8] This is of course not the case in reality. Nevertheless, this assumption is kept throughout the rest of this chapter to enable the initial scenes being drawn from the dataset. An optimization of initial scenes when they are purely based on sampling from the dataset is not possible since a pure dataset does not contain descriptive optimizable parameters. The optimization would require an initial scene model. It is possible to neglect this by including a *relaxation phase* (section 5.3.3). The challenges arising from this concept are not intrinsic to the proposed algorithmic scheme. They are finally resolved in chapters 6 and 7 by the introduction of initial scene models.

by

$$p_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}}\right) = \overbrace{p_{0,\text{irr.},1}\left(\varphi_{\boldsymbol{x}_0}\right) \cdot p_{0,\text{irr.},2}\left(\varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right)}^{p_{0,\text{irr.}}\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right)} \cdot p_{0,\text{rel.}}\left(\varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right). \tag{5.42}$$

The distribution $p_{0,\text{irr.},1}$ of initial scenes is, as already mentioned, described by drawing arbitrary samples from the highD dataset (see fig. B.3 for an exemplary sample). This makes sure that realistic scenes which are not limited to certain scenario classes or complexities are used. For simplicity, the lateral positions within a lane and the lateral velocities of the vehicles are neglected[9]. On the German Autobahn there is always a constructional median barrier. Therefore vehicles driving in the opposite direction are totally decoupled and only the vehicles traveling in the same direction as the ego vehicle are relevant for its safety. Hence, when sampling from the highD dataset only these vehicles are extracted.

After having drawn a scene from the dataset, one of the vehicles in the initial scene is declared to be the ego vehicle which is controlled by the ADF model given in section 5.3.1.2. It is the simulation's aim to test this ego vehicle/ADF. The rest of the vehicles are governed by the behavior models described in section 5.3.1.3. The ADF model and the behavior models together then account for $p_{\text{beh.}}\left(\varphi_{\boldsymbol{x}_1}, ..., \varphi_{\boldsymbol{x}_{n_{\varphi_\xi}}} \middle| \varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}_{\text{beh.}}}\right)$ as required by (5.18) and (5.19).

Additionally, it is $\boldsymbol{\phi} = \boldsymbol{\phi}_{\text{beh.}}$. That means all of the non-physical parameter in the used simulation framework are behavior parameters which determine the vehicle dynamics, no further parameters are required. The parameters in $\boldsymbol{\phi}_{\text{beh.}}$ are determined by the used behavior models (IDM and MOBIL). They are discussed in section 5.3.1.3. Since the behavior parameters shall characterize the intrinsic properties of the drivers of the single vehicles, each vehicle gets its own set of behavior parameters. It is assumed that the behavior parameters do not change during the considered time horizon n_{φ_ξ} of the simulations. For the sake of simplicity, the non-physical parameters are sampled independently from each other. That

[9] This is also necessary to prevent the initial scenes which cannot be optimized to have a dominating share in the causation of critical scenes. It is therefore the *relaxation phase*'s (which only relaxes longitudinal parts of the initial scenes, see section 5.3.3) equivalent for lateral parts of a scene. Hence, to include the lateral parts an initial scene model which sufficiently addresses the lateral parts of a scene is required.

is, both $p_{0,\text{rel.}}$ and $p_{0,\text{irr.},2}$ are fully factorized distributions with factors of truncated Cauchy distributions

$$p^{(\mathfrak{p})}_{\text{cauchy,trunc}}\left(\varphi_{\mathfrak{p}}\middle|\mathbf{u}^{(\mathfrak{p})},\mathfrak{B}_{\mathfrak{p}}\right)=\begin{cases}\frac{1}{z_{\mathfrak{B}_{\mathfrak{p}}}\pi}\frac{u_2^{(\mathfrak{p})}}{\left(u_2^{(\mathfrak{p})}\right)^2+\left(\varphi_{\mathfrak{p}}-u_1^{(\mathfrak{p})}\right)^2} & \text{if } \varphi_{\mathfrak{p}}\in\mathfrak{B}_{\mathfrak{p}}\\ 0 & \text{else}\end{cases}. \tag{5.43}$$

The random variable $\mathfrak{p}$ declares the parameters of IDM and MOBIL which are introduced in section 5.3.1.3 and contained in $\boldsymbol{\phi}_{\text{rel.}}$ respectively $\boldsymbol{\phi}_{\text{irr.}}$. The parameters in the 2×1 matrix $\mathbf{u}^{(\mathfrak{p})}=\begin{pmatrix}u_1^{(\mathfrak{p})} & u_2^{(\mathfrak{p})}\end{pmatrix}^{\dagger}$ give the position of the peak and the width of the Cauchy distribution. The support of the truncated Cauchy distribution is limited to the interval $\mathfrak{B}_{\mathfrak{p}}$. The values $\mathbf{u}^{(\mathfrak{p})}$ as well as $\mathfrak{B}_{\mathfrak{p}}$ are enumerated in table 5.2. The parameter $z_{\mathfrak{B}_{\mathfrak{p}}}$ renorms the distribution to the truncation interval. The Cauchy distributions are centered on the standard values $u_1^{(\mathfrak{p})}$ of the parameters. Note, that all the behavior parameters get sampled separately for each vehicle since they describe the intrinsic properties of a driver and there is obviously an extra, different driver in each vehicle. The Cauchy distribution was used because its slope converges relatively slowly to 0. Since these slopes often contain the critical and thereby interesting simulation runs, a product of Cauchy distributions seems to be a promising starting point for a challenging simulation framework with rare critical events. In contrast, when using Gaussian distributions, the slopes would vanish pretty fast. This could cause an underflow in the floating-point arithmetic of a computer and thereby the loss of events from the slopes of the distributions.

5.3.1.2 Ego vehicle

The vehicle on the middle lane in a generated initial scene whose distance to the longitudinal middle point of the simulated highway section is minimal, is defined to be the ADF driven ego vehicle. This ego vehicle shall be tested by the simulation.

The ego vehicle's dynamics are governed by a simple adaptive cruise control (ACC) model as introduced by Gelder and Paardekooper [30]. This means, that the longitudinal acceleration of the ego vehicle is controlled by

$$a_{\text{ego}}=\min\left(u_{\text{CC}},u_{\text{ACC}}\right), \tag{5.44}$$

Table 5.1 The parameters used for the ego vehicle controller. Their values are taken from [30]. v_r is the desired reference velocity, k_{VCC} gives the velocity gain of the cruise controller, τ_{h} is the desired time headway and s_0 describes the desired minimal safety distance. The parameters k_{d1} and k_{d2} correspond to the distance gains of the ACC at high and low velocity. The parameter σ_d describes how fast k_{d2} changes to k_{d1} if the velocity increases. The parameter k_{v} is called velocity difference gain. The table was also published by Jesenski et al. [134]; © 2020 IEEE.

Parameter	Value	Parameter	Value
v_r	$36\,\frac{\mathrm{m}}{\mathrm{s}}$	k_{d1}	$0.7\,\mathrm{s}^{-2}$
k_{VCC}	$1.3\,\mathrm{s}^{-1}$	k_{d2}	$2.0\,\mathrm{s}^{-2}$
τ_{h}	$2.0\,\mathrm{s}$	σ_d	$5.0\,\frac{\mathrm{m}}{\mathrm{s}}$
s_0	$1.54\,\mathrm{m}$	k_{v}	$0.35\,\mathrm{s}^{-1}$

with

$$u_{\mathrm{CC}} = k_{\mathrm{VCC}}\left(v_r - v_{\mathrm{ego}}\right), \tag{5.45}$$

$$u_{\mathrm{ACC}} = k_{\mathrm{d}}(v_{\mathrm{ego}})(d_{\mathrm{ego}} - \tau_h v_{\mathrm{ego}} - s_0) + k_{\mathrm{v}}\dot{d}_{\mathrm{ego}}, \tag{5.46}$$

$$k_{\mathrm{d}}(v) = k_{\mathrm{d1}} + (k_{\mathrm{d2}} - k_{\mathrm{d1}}) \cdot e^{-\frac{v_{\mathrm{ego}}^2}{2\sigma_d}}. \tag{5.47}$$

Table 5.1 contains the parametrization of the ACC. The ego vehicle's acceleration a_{ego} is confined by the interval $I_{\mathrm{ego}} = \left[-8\,\mathrm{m/s}^2, 8\,\mathrm{m/s}^2\right]$. The ego vehicle's velocity is given by v_{ego}, d_{ego} is the distance to its predecessor. The desired speed v_r is of course dependent on the simulated location. In the presented case v_r is derived by the advisory speed limit of $130\,\frac{\mathrm{km}}{\mathrm{h}}$. In the presented simulation framework optimal perception is assumed. Sensor noise could however easily be added to the simulations.

5.3.1.3 Behavior models for surrounding vehicles

In order to model the stochastic time dynamics expressed by $p_{\mathrm{beh.}}$, probabilistic behavior models (BMs) are needed to control the agents which surround the ADF-controlled ego vehicle. Since the main interest of this work is to show the applicability of the IS strategy to large unstructured scenarios, there is no focus on detailed vehicle behavior. Hence, simple parametric BMs are sufficient for the purposes of this work. A heavily used model for the longitudinal dynamics of an agent is the intelligent driver

model (IDM) [104] which shall now be applied. For the lateral dynamics, *minimizing overall braking induced by lane changes* (MOBIL) [52] is utilized.

The IDM which governs the longitudinal dynamics of a vehicle is defined by

$$\dot{v}_t^{(\mathrm{det})}(v_{t-t_{\mathrm{react}}}, \Delta v_{t-t_{\mathrm{react}}}, d_{t-t_{\mathrm{react}}}) = A \left[1 - \left(\frac{v_{t-t_{\mathrm{react}}}}{V_0} \right)^{\delta} - \left(\frac{s^* \left(v_{t-t_{\mathrm{react}}}, \Delta v_{t-t_{\mathrm{react}}} \right)}{d_{t-t_{\mathrm{react}}}} \right)^2 \right] \tag{5.48}$$

with the *desired minimum gap*

$$s^*(v, \Delta v) = S_0 + S_1 \cdot \sqrt{\frac{v}{V_0}} + T \cdot v + \frac{v \cdot \Delta v}{2\sqrt{A \cdot B}}. \tag{5.49}$$

The parameters of the IDM are described in table 5.2. Note that the original model proposed by Treiber et al. [104] was modified by adding a reaction time t_{react}. The deterministic acceleration $\dot{v}_t^{(\mathrm{det})}$ for time step t is dependent on the three variables $\left(v_{t-t_{\mathrm{react}}}, \Delta v_{t-t_{\mathrm{react}}}, d_{t-t_{\mathrm{react}}}\right)$, which are the velocity of the agent, the velocity difference to its predecessor and the distance to its predecessor at time $t - t_{\mathrm{react}}$. On the German Autobahn it is in general forbidden to overtake another vehicle on the right. To implement this rule, the minimization

$$\dot{v}_{t,*}^{(\mathrm{det})} = \begin{cases} \min\left(\dot{v}_t^{(\mathrm{det})}, \dot{v}_t^{(\mathrm{det,left})} \right) & \text{if } v_{t-t_{\mathrm{react}}} > v_{\mathrm{lead}} > v_{\mathrm{c}} \\ \dot{v}_t^{(\mathrm{det})} & \text{otherwise} \end{cases} \tag{5.50}$$

as proposed in [52] is implemented. $\dot{v}_t^{(\mathrm{det,left})}$ is the agent's IDM acceleration when the distance and velocity difference to its predecessor on the left lane are given as inputs. v_{lead} is the velocity of the predecessor on the left lane and v_{c} is the maximum velocity for which it is allowed to overtake on the right, e.g. in congestions. On the German Autobahn it is $v_{\mathrm{c}} = 60\ \frac{\mathrm{km}}{\mathrm{h}}$.

Since the IDM is deterministic and the simulation shall be probabilistic, the Gaussian distribution

$$\dot{v}_t^{(\mathrm{prob})} \sim p(a) = \frac{1}{\sqrt{2\pi\sigma^2}} \mathrm{e}^{-\frac{\left[a - \dot{v}_{t,*}^{(\mathrm{det})}\right]^2}{2\sigma^2}}. \tag{5.51}$$

Table 5.2 Parameters $\boldsymbol{\phi}_i \subseteq \boldsymbol{\phi}_{\text{beh.}}$ needed to specify the BMs of a surrounding agent i. The parameters are sampled as defined in section 5.3.1.1. The parameters with "-" in the fourth column are not sampled, but always set to the values in the third column which are given instead of interval limits. The parameters in the fourth column give the initial peak position and the width of the Cauchy distributions in (5.43). The peak values of the parameters are mostly oriented by the values given by Kesting et al. [52] and Treiber et al. [104]. The table is based on table 3 published by Jesenski et al. [134]; © 2020 IEEE.

	Param $\mathfrak{p}$	Description	Interval $\mathfrak{B}_\mathfrak{p}$	$\left(\mathbf{u}^{(\mathfrak{p})}\right)^\dagger$
IDM	V_0	desired velocity in $\frac{\text{m}}{\text{s}}$	$[0.001, 70]$	$\begin{pmatrix}36 & 2.5\end{pmatrix}$
	T	safe time headway in s	$[0.2, 2.5]$	$\begin{pmatrix}1.6 & 0.2\end{pmatrix}$
	A	max. acceleration in $\frac{\text{m}}{\text{s}^2}$	$[0.2, 1.5]$	$\begin{pmatrix}0.73 & 0.1\end{pmatrix}$
	B	des. deceleration in $\frac{\text{m}}{\text{s}^2}$	$[0.5, 3]$	$\begin{pmatrix}1.67 & 0.8\end{pmatrix}$
	δ	acc. exponent	$[4 \pm 0.001]$	$\begin{pmatrix}4 & 0.01\end{pmatrix}$
	S_0	jam distance 1 in m	$[1, 3]$	$\begin{pmatrix}2 & 1\end{pmatrix}$
	S_1	jam distance 2 in m	$[0, 3]$	$\begin{pmatrix}0 & 1\end{pmatrix}$
	t_{react}	reaction time in s	$[0, 4]$	$\begin{pmatrix}0.001 & 0.1\end{pmatrix}$
	v_{c}	congest. velocity in $\frac{\text{m}}{\text{s}}$	16	-
	σ	std. deviation in $\frac{\text{m}}{\text{s}^2}$	$[0.2, 1.4]$	$\begin{pmatrix}0.75 & 0.3\end{pmatrix}$
	a_{limits}	accel. limits in $\frac{\text{m}}{\text{s}^2}$	-10 and 10	-
MOBIL	$\omega_{\text{pol.}}$	politeness factor	$[0, 1]$	$\begin{pmatrix}0.9 & 0.1\end{pmatrix}$
	Δa_{th}	changing threshold in $\frac{\text{m}}{\text{s}^2}$	0.2	-
	b_{min}	min. safe dec. in $\frac{\text{m}}{\text{s}^2}$	$[-2, 2]$	$\begin{pmatrix}0.6 & 0.3\end{pmatrix}$
	Δa_{bias}	bias for right lane in $\frac{\text{m}}{\text{s}^2}$	0.3	-

is applied to get the final probabilistic acceleration $\dot{v}_t^{(\text{prob})}$.

The Gaussian function is truncated (and renormed) by the interval $a_{\text{limits}} = \left[-10\,\frac{\text{m}}{\text{s}^2}, 10\,\frac{\text{m}}{\text{s}^2}\right]$ and the agents are prevented to drive in the wrong direction by setting $v \geq 0\,\frac{\text{m}}{\text{s}}$.

MOBIL Since a multi-lane highway section shall be simulated, it is necessary that the lateral dynamics of the agents are determined. For that, MOBIL [52] is used. MOBIL works by calculating the agents' lane change decisions. In order to do so, the benefit of a lane change to the right is

Table 5.3 Conditions for lane changes. The table was already published by Jesenski et al. [134]; © 2020 IEEE.

Conditions	Decision
$\tau_R > \tau_L \wedge \tau_R > 0 \wedge \tau_{\text{safe}}^{(\text{right})}$	change to the right
$\tau_L > \tau_R \wedge \tau_L > 0 \wedge \tau_{\text{safe}}^{(\text{left})}$	change to the left
otherwise	no lane change

defined by

$$\tau_R = \tilde{a} - a + \omega_{\text{pol.}} \left(\tilde{a}_o - a_o\right) - \left(\Delta a_{\text{th}} - \Delta a_{\text{bias}}\right). \tag{5.52}$$

The benefit of a lane change to the left is analogously defined by

$$\tau_L = \tilde{a} - a + \omega_{\text{pol.}} \left(\tilde{a}_n - a_n\right) - \left(\Delta a_{\text{th}} + \Delta a_{\text{bias}}\right). \tag{5.53}$$

$\tilde{a}$ and a are the longitudinal IDM accelerations of the ego vehicle after and before the hypothetic lane change. Analogously, $\tilde{a}_{o/n}$ and $a_{o/n}$ are the IDM accelerations of the agent's successor on its old/new lane after and before the lane change. The parameter $\omega_{\text{pol.}}$ decides about the politeness of the lane changing vehicle, that means how strong it considers the effects of the lane change on relevant surrounding traffic participants. The value for Δa_{th} is the overall advantage a lane change minimally must achieve to be executed and Δa_{bias} realizes the keep-right directive as required in most European countries. As also proposed by Kesting et al. [52] an additional safety criterion,

$$\tau_{\text{safe}} = \tilde{a} > b_{\min} \wedge \tilde{a}_n > b_{\min} \tag{5.54}$$

is used. This makes sure that no vehicle has to brake stronger than allowed by the *minimum safe deceleration* $b_{\min}$ after the lane change has taken place. The lane change decision itself is based upon the conditions in table 5.3. When the lane change decision was made, the lateral behaviour is determined using constant accelerations. These are applied to reach a desired lateral lane change velocity and to brake afterwards. The used values for the described MOBIL parameters are also given in table 5.2.

5.3.2 Selection of causality groups

The proposed family of distributions $q_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}} \middle| \varphi_{\mathfrak{v}}\right)$ is chosen analogously to the unskewed distribution described in section 5.3.1.1. Consequently,

(5.21) is modified into

$$q_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}} \middle| \varphi_{\mathfrak{v}}\right) = \overbrace{p_{0,\text{irr.},1}\left(\varphi_{\boldsymbol{x}_0}\right) \cdot p_{0,\text{irr.},2}\left(\varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right)}^{p_{0,\text{irr.}}\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right)} \cdot q_{0,\text{rel.}}\left(\varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\mathfrak{v}}\right). \tag{5.55}$$

The distributions $p_{0,\text{irr.},1}\left(\varphi_{\boldsymbol{x}_0}\right)$ and $p_{0,\text{irr.},2}\left(\varphi_{\boldsymbol{\phi}_{\text{irr.}}}\right)$ were already described in section 5.3.1.1. The remaining distribution $q_{0,\text{rel.}}$ is split into causality groups as described in (5.22). For maximal efficiency gains, the causality groups $\mathfrak{C}$ are chosen to consist of single parameters. Therefore, $q_{0,\text{rel.}}$ is chosen to be factorized by truncated Cauchy mixture distributions

$$q_{0,\text{rel.},\mathfrak{p}}\left(\varphi_{\mathfrak{p}} \middle| \varphi_{\mathfrak{v}^{(\mathfrak{p})}}\right) = \sum_{i=1}^{3} \varphi_{m_i^{(\mathfrak{p})}} \cdot p^{(\mathfrak{p})}_{\text{cauchy,trunc}}\left(\varphi_{\mathfrak{p}} \middle| \varphi_{\mathfrak{v}_i^{(\mathfrak{p})}}, \mathfrak{B}_{\mathfrak{p}}\right) \tag{5.56}$$

for $\mathfrak{p} \in \mathfrak{C} = \boldsymbol{\phi}_{\text{rel.}}$. $q_{0,\text{rel.},\mathfrak{p}}$ is defined by 9 optimizable parameters since $\mathfrak{v}^{(\mathfrak{p})} = \bigcup_{i=1}^{3}\left(\left\{m_i^{(\mathfrak{p})}\right\} \cup \mathfrak{v}_i^{(\mathfrak{p})}\right)$. The interval $\mathfrak{B}_{\mathfrak{p}}$ is kept constant at the values given in table 5.2. The boundary condition $\sum_{i=1}^{3} \varphi_{m_i^{(\mathfrak{p})}} = 1$ must hold. The truncated Cauchy distributions $p^{(\mathfrak{p})}_{\text{cauchy,trunc}}\left(\varphi_{\mathfrak{p}} \middle| \varphi_{\mathfrak{v}_i^{(\mathfrak{p})}}, \mathfrak{B}_{\mathfrak{p}}\right)$ are defined as given by (5.43).

Consequently, by setting the products of (5.43) and (5.56) into (5.25), the relation between unskewed and skewed distribution ends up being

$$\mathcal{K}_{\text{rel.}}\left(\varphi_h; \boldsymbol{\omega}\right) = \prod_{\mathfrak{p} \in \boldsymbol{\phi}_{\text{rel.}}} \frac{p^{(\mathfrak{p})}_{\text{cauchy,trunc}}\left(\varphi_{\mathfrak{p}} \middle| \mathfrak{u}^{(\mathfrak{p})}, \mathfrak{B}_{\mathfrak{p}}\right)}{\sum_{i=1}^{3} \varphi^{(\boldsymbol{\omega})}_{m_i^{(\mathfrak{p})}} \cdot p^{(\mathfrak{p})}_{\text{cauchy,trunc}}\left(\varphi_{\mathfrak{p}} \middle| \boldsymbol{\omega}_i^{(\mathfrak{p})}, \mathfrak{B}_{\mathfrak{p}}\right)}. \tag{5.57}$$

When adapting the sampling distribution in the iterative optimization procedure as introduced in section 5.2.2.2, the relation of course must additionally be adjusted. For the executed simulations in section 5.3.4, $\boldsymbol{\phi}_{\text{rel.}}$ has empirically been selected to consist of specific parameters of specific vehicles as listed in table 5.4. The relevant vehicle positions are illustrated in fig. 5.2 .

Note, that the parameters $\mathfrak{p}$ are chosen and optimized in a position-dependent and not a vehicle-dependent manner. That means, when a

Table 5.4 The parameters $\boldsymbol{\phi}_{\text{rel.}}$ which are used to optimize the pdf. The set $\boldsymbol{\phi}_i \subseteq \boldsymbol{\phi}_{\text{beh.}}$ contains the IDM and MOBIL parameters of vehicle i, however only selected parameters of each vehicle are included in the set of relevant parameters. Compare table 5.2 for the meaning of the parameters. For the vehicles' nomenclature see fig. 5.2.

Vehicle i	$\boldsymbol{\phi}_i \setminus \boldsymbol{\phi}_{\text{irr.}}$
$L_{\text{same},0}$	V_0
$L_{\text{same},1}$	V_0
$T_{\text{same},0}$	t_{react}

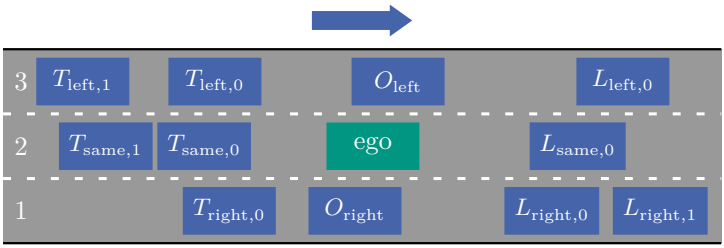


Figure 5.2 The position-dependent nomenclature of vehicles in relation to the the ego vehicle. The index iterates with growing distance to the ego vehicle.

vehicle changes its position in relation to the ADF-based ego vehicle, its behavior parameters get changed to the respectively assigned values.

This position-dependency can be justified by the fact, that critical behavior of a vehicle is strongly determined by its position and therefore the parametric family of distributions $q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right)$ should be chosen accordingly. As an example, a vehicle driving behind the ego vehicle is probably more critical if it accelerates fast and has large reaction times. However, the vehicle preceding the ego should in contrast decelerate fast and drive slower in order to provoke critical scenarios. The introduced procedure is not limited to position-dependent sampling. It could easily be used to sample vehicle-dependent.

With the chosen relevant parameters and an exemplary scene with $n_{\text{veh}}^{(\varphi_\xi)} = 25$ vehicles, the separation into relevant and irrelevant parameters and the introduction of causality groups would enable the reduction of one optimization with $n_{\text{veh}}^{(\varphi_\xi)} \cdot |\boldsymbol{\phi}_i| \cdot 9 = 2475$ dimensions to 3 optimizations with 9 dimensions (compare section 5.2.1).

5.3.3 Relaxation phase

As already explained, the initial scenes $\boldsymbol{x}_0$ are simply drawn from the highD dataset which makes it impossible to influence these initial scenes by optimizing the set $\boldsymbol{\phi}_{\text{rel.}}$ via the CE method. This introduces the implicit assumption that the initial scene is not relevant for the criticality of a total scenario into the design of $q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right)$. Obviously, this cannot be true in reality since for example the initial distances and relative velocities between neighboring vehicles will be important for a scenario's criticality. Mathematically this implies that the proposed family of distributions $q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right)$ has a certain deviation to the optimal IS distribution (5.12) which it cannot undercut no matter how the parameters $\mathfrak{v}$ are chosen. As a result the standard deviation (5.11) might not be reduced as far as desired. It has turned out in the experiments that this case typically leads to an underestimation of the collision rate since not all critical parts $\boldsymbol{\varepsilon}$ of reality were found and thereby the probability values, the distribution $q\left(\varphi_h \middle| \mathfrak{v}^*\right)$ allocates to its known critical scenarios in $\boldsymbol{\varepsilon}$ are too large on average. Therefore, the relation in (5.7) for the drawn samples is on average too low. However, (5.10) shows that the expectation value obtained from IS should always be correct. This is consistent with the described situation since critical runs $\xi \in \boldsymbol{\varepsilon}$ which could not be optimized and emphasized by the found distribution $q\left(\varphi_h \middle| \mathfrak{v}^*\right)$ will have too large relations $\mathcal{K}_*\left(\varphi_h\right)$ since their occurrence is underweighted in $q\left(\varphi_h \middle| \mathfrak{v}^*\right)$. If these samples are found and sampled, this introduces a large increase in the estimation of the collision rate since $\mathcal{K}_*\left(\varphi_h\right)$ becomes very large in these cases. However since they are not emphasized by the optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}^*\right)$ they will rarely be sampled. Consequently, the number of samples would have to be strongly increased in order to find enough of these "increases" in order to get a good estimation of the collision rate.

Since this chapter shall give a proof of concept that the proposed method can handle large scale scenes as included in the highD dataset, it is sufficient to handle the described problem by reducing the influence of the initial scene on the scenario's criticality. This can be achieved by the introduction of a *relaxation phase*[10], which allows decoupling the initial scene from the

[10] This relaxation phase of course changes the unskewed version of the simulation. The calculated collision rates will also change undoubtedly. However, the aim of this work is not to prove the real collision rates of a system, but to give a prove of concept of the introduced method. An altered unskewed simulation does not pose a problem in this context.

Table 5.5 Parameters used in algorithm 1 for both optimizations. The metric $g_{\text{crit.}}^{(\text{RSS Dist})}$ is the metric based on $g_{\text{pair}}^{(\text{RSS Dist})}(p,t,\xi)$ defined in section 4.4.2, whereas $g_{\text{crit.}}^{(\text{RSS Beh})}$ is based on $g_{\text{pair}}^{(\text{RSS Beh})}(p,t,\xi)$ given in section 4.4.3. Both metrics base on RSS which was in this relation parameterized by the values in table 5.6.

$g_{\text{crit.}}$	$\mathfrak{N}$	η	$k_{\max}$	γ	β_1	β_2	z_0	z_1	z_2	k_{add}	$\mathfrak{N}_{\text{add}}$
$g_{\text{crit.}}^{(\text{RSS Dist})}$	10 000	0.02	100	−1	0.2	0.3	2	0.96	0.25	2	20 000
$g_{\text{crit.}}^{(\text{RSS Beh})}$	10 000	0.02	100	−8	0.2	0.3	2	0.96	0.25	2	20 000

set $\boldsymbol{\varepsilon}$. During this relaxation phase, which lasts for the 40 first steps of a simulation run, no criticality assessment takes place. Additionally, some parameters of the vehicles in a simulation run are changed during the relaxation phase: It is for example $t_{\text{react}} = 0\,\text{s}$ for all vehicles. Besides, the vehicles' values for V_0 are set to their velocities in the initial scene. S_0 and T of all vehicles are set to the standard values of IDM. The Gaussian noise σ is reduced to 0.01 and the acceleration limits of the vehicles are enlarged to $-30\,\frac{\text{m}}{\text{s}^2}$ and $30\,\frac{\text{m}}{\text{s}^2}$. Last but not least, after the end of the relaxation phase, a vehicle will always at least have knowledge about the scene directly after the end of the relaxation phase, no matter if the elapsed time since the relaxation phase's end is smaller than the vehicle's reaction time t_{react}. These changes allow the initial scene to relax into a safer state before critical states are again produced by the optimizable behavior models. Note however, that the necessity of a relaxation phase is not a general problem of the described approach. It will be shown in chapter 7 that the introduction of an optimizable initial scene model based on the techniques mentioned in chapter 6 renders the relaxation phase unnecessary.

5.3.4 Evaluation

The parametric family of functions $q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right)$ based on the causality groups given in section 5.3.2 is now optimized towards the optimal IS distribution (5.12) by the iterative scheme in algorithm 1, with the additions introduced in section 5.2.2. The optimization is executed separately on $g_{\text{crit.}}^{(\text{RSS Dist})}$ which is a purely physical metric and on $g_{\text{crit.}}^{(\text{RSS Beh})}$ which also assesses the behavior of the ADF controlled ego vehicle. For both metrics, it is $\boldsymbol{\mathcal{S}} = \{\text{ego}\}$, compare chapter 4. The parameters used for both optimizations are summarized in tables 5.5 and 5.6. The simulations are run with time

Table 5.6 The numerical values for the RSS parameters defined in table 4.1 which were used to define both RSS-based safety metrics.

Parameter	Value	Parameter	Value
ρ	$0.5\,\mathrm{s}$	$a_{\mathrm{max,accel}}$	$4\,\frac{\mathrm{m}}{\mathrm{s}^2}$
$a_{\mathrm{min,brake}}$	$7\,\frac{\mathrm{m}}{\mathrm{s}^2}$	$a_{\mathrm{max,brake}}$	$7\,\frac{\mathrm{m}}{\mathrm{s}^2}$
$a_{\mathrm{lat,max,accel}}$	$1\,\frac{\mathrm{m}}{\mathrm{s}^2}$	$a_{\mathrm{lat,min,brake}}$	$2\,\frac{\mathrm{m}}{\mathrm{s}^2}$
μ	$0.04\,\mathrm{m}$		

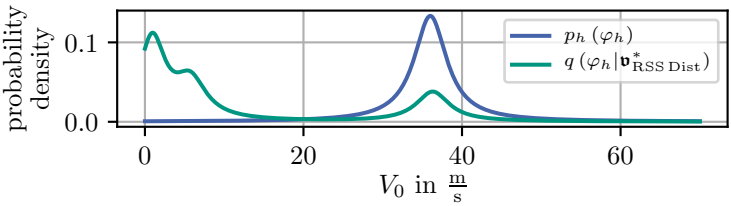
steps $\Delta t = 0.04\,\mathrm{s}$ and a time horizon of $n_{\varphi_\xi} = 300$ steps.

5.3.4.1 Evaluation on physics-based metric

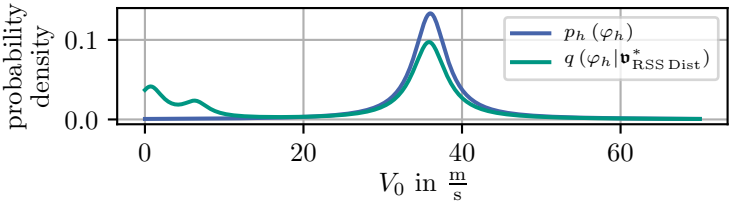
The optimization on $g_{\mathrm{crit.}}^{(\mathrm{RSS\ Dist})}$ converged to $\gamma = -1$ after 3 iterations. As already shown in table 5.5, two additional optimization steps follow. The resulting, optimized parameter set is denoted by $\mathfrak{v}^*_{\mathrm{RSS\ Dist}}$. The consequent optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}^*_{\mathrm{RSS\ Dist}}\right)$ emphasizes the critical event set

$$\boldsymbol{\varepsilon}_{\mathrm{RSS\ Dist}} = \left\{ \varphi_\xi \in \boldsymbol{\Omega} : g_{\mathrm{crit.}}^{(\mathrm{RSS\ Dist})}\left(\varphi_\xi, \boldsymbol{\mathcal{S}}\right) = -1 \right\}. \tag{5.58}$$

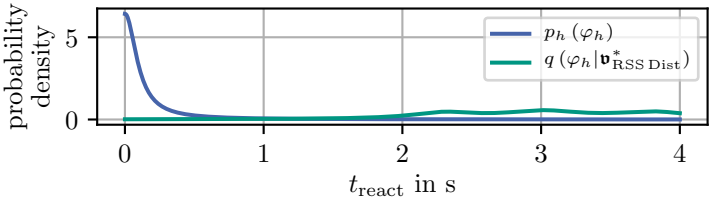
By definition, this set comprises all physical collisions with participation of the ADF-controlled ego vehicle. Figure 5.3 visualizes the effects of the optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}^*_{\mathrm{RSS\ Dist}}\right)$ by displaying $q\left(\varphi_h \middle| \mathfrak{v}^*_{\mathrm{RSS\ Dist}}\right)$ and the original distribution $p_h\left(\varphi_h\right)$ marginalized on the relevant parameters in the causality groups. It is obvious, that all three causality group parameters are strongly shifted. The shifts can be understood intuitively. For both velocity parameters $(L_{\mathrm{same},0}, V_0)$ and $(L_{\mathrm{same},1}, V_0)$, small values are strongly emphasized in the optimized distribution. The reason is that low values of IDM's desired velocity V_0 expresses the desire of the respective vehicle to drive slowly. The slower a vehicle desires to drive, the higher the probability, that the respective vehicle brakes, becomes. Of course, the braking of a leading vehicle like $L_{\mathrm{same},0}$ or $L_{\mathrm{same},1}$ increases the criticality of a scene and the likelihood of a collision taking place. For the trailing vehicle $T_{\mathrm{same},0}$, the situation is a little bit different. If the trailing vehicle drives slower, this would decrease a scenario's criticality since the trailing vehicle tends to drive slower than the ego vehicle and the distance between them would increase. Hence, V_0 is not of interest here. However, if the reaction time



(a) $L_{\text{same},0}$



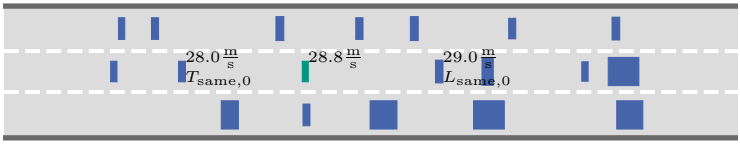
(b) $L_{\text{same},1}$



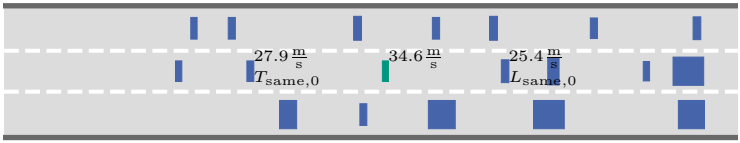
(c) $T_{\text{same},0}$

Figure 5.3 Comparison of $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist}}\right)$ and $p_h\left(\varphi_h\right)$. In detail, the images show a comparison of the marginals of the relevant behavior parameters/causality groups. **(a)** gives the marginal of the desired velocity V_0 of vehicle $L_{\text{same},0}$, **(b)** shows the same for $L_{\text{same},1}$ and **(c)** gives the reaction time t_{react} of the trailing vehicle $T_{\text{same},0}$.

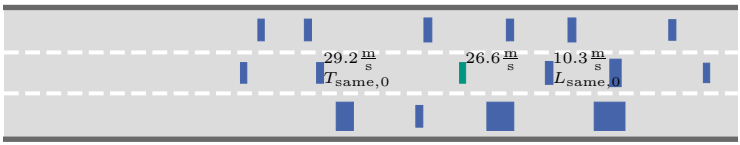
t_{react} of the trailing vehicle increases, then its capability to react on ego's dynamics decreases which increases the probability that it crashes into it. Consequently, as shown in fig. 5.3(c), the optimization emphasizes large values of t_{react}. Note, that the shift in the distribution of $L_{\text{same},1}$ is weaker than the shift in the distribution of $L_{\text{same},0}$. Hence, a vehicle's effect on ego's criticality decreases when its separation to ego increases. This strengthens the confidence in the concept of causality groups since that means a lot of simulation parameters can be neglected during optimization.



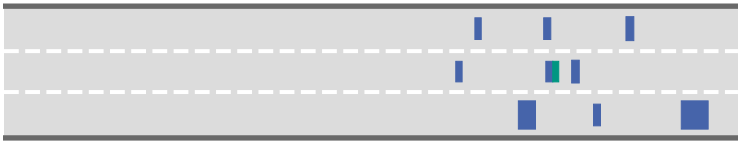
(a) $t = 0\,\text{s}$ after the start of the simulation.



(b) $t = 1.6\,\text{s}$ after the start of the simulation.



(c) $t = 3.2\,\text{s}$ after the start of the simulation.



(d) $t = 8.72\,\text{s}$ after the start of the simulation.

Figure 5.4 Illustration of a highly critical simulation run. The ADF controlled ego vehicle is shown in green, the surrounding vehicles are blue. The vehicles appear distorted since the road section has a length of about 480 m and a width of about 13 m, the relation of the images' width and height is therefore not appropriate. **(a)** displays the initial scene drawn from the highD dataset. **(b)** shows the scene at $t = 1.6\,\text{s}$. This is directly after the end of the relaxation phase. Afterwards, the scene quickly becomes dangerous. A low value for the desired velocity V_0 now causes the predecessor $L_{\text{same},0}$ to brake hard. In **(c)**, it is already driving very slowly at $10.3\,\frac{\text{m}}{\text{s}}$. In **(d)** the successor vehicle $T_{\text{same},0}$ crashes into the ego vehicle. Due to a large reaction time, it could not react quickly enough to the decelerating ego vehicle which had to brake in order to prevent crashing into $L_{\text{same},0}$.

Figure 5.4 illustrates a typical critical simulation run which was drawn by $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist}}\right)$. Obviously, the already mentioned effects provoke a

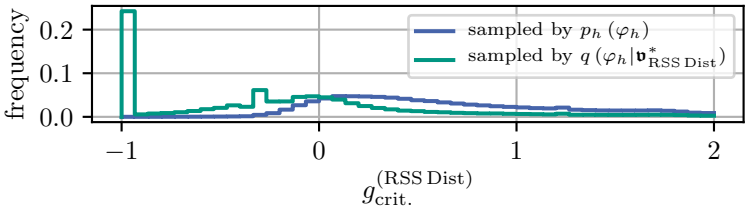


Figure 5.5 Criticality histogram of the optimized distribution in comparison with the original unskewed one. The histograms are based on 120 000 samples drawn from $q\left(\varphi_h\middle|\mathfrak{v}^*_{\text{RSS Dist}}\right)$ and 650 000 samples drawn from $p_h\left(\varphi_h\right)$.

crash between $T_{\text{same},0}$ and ego.

The optimized distribution can further be evaluated by comparing the criticality values produced by drawing from it with the criticality values when drawing from the original distribution $p_h\left(\varphi_h\right)$. The resulting histograms are displayed in fig. 5.5. The histograms show that the optimized distribution generates much more critical scenarios. More than 20 % of all runs generated with the optimized distribution are located in the worst bin directly next to a collision at the metric value $g_{\text{crit.}}^{(\text{RSS Dist})} = -1$. In contrast, when sampling from the original distribution $p_h\left(\varphi_h\right)$, most of the simulation runs were uncritical at a metric value larger 0 (compare section 4.4.2).

This high share of critical sampled scenes allows extracting more information about an ADF's safety with fewer simulation runs. This is illustrated by fig. 5.6. The application of (5.9) on the values sampled by the optimized distribution causes a significantly faster convergence of the approximated collision rate than when naively applying (5.4) on the samples from the original distribution. The approximated collision rate of the discussed ADF lies at approximately

$$P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}_{\text{RSS Dist}}\right) \approx 5.98 \times 10^{-5}. \tag{5.59}$$

Figure 5.6 shows that the standard deviations of the original distribution are larger in general. When comparing them numerically, it is found, that the standard deviation of sample 650 000 of $p_h\left(\varphi_h\right)$'s curve is equivalent to the standard deviation at sample 6086 of $q\left(\varphi_h\middle|\mathfrak{v}^*_{\text{RSS Dist}}\right)$'s curve. This corresponds to an acceleration factor of

$$\mathfrak{F}_{\text{accel.}}^{(\text{RSS Dist})} \approx 106.8. \tag{5.60}$$

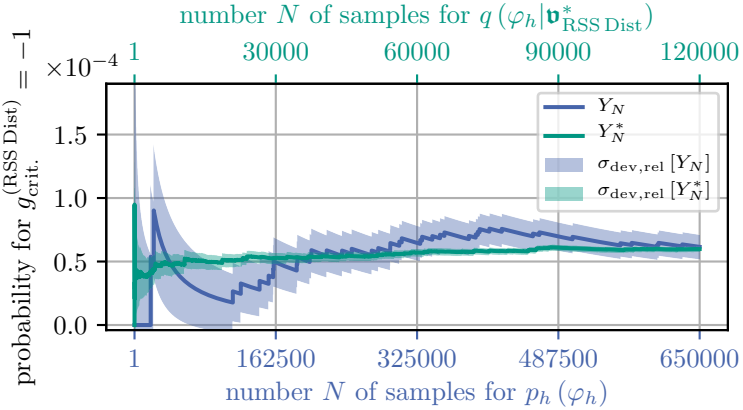


Figure 5.6 The occurrence rate of $\varepsilon_{\text{RSS Dist}}$ which corresponds to collisions of the ego vehicle. The occurence rate is determined both by sampling from the original and the optimized distributions. Additionally, the standard deviations of both methods are drawn in.

In order to check if the purely physical metric $g^{(\text{RSS Dist})}_{\text{crit.}}$ is sufficient, all simulation runs with $g^{(\text{RSS Dist})}_{\text{crit.}} = -1$ are extracted from the 120 000 samples drawn from the optimized distribution. About 28 500 relevant simulation runs are obtained this way. The metric values of $g^{(\text{RSS Beh})}_{\text{crit.}}$ of these runs are calculated and shown in a histogram in fig. 5.7. As the figure shows, most of the simulation runs, which are critical for the purely physics-based metric, are not in the critical range for the behavior-based metric. Therefore, for the most of these cases, the ADF-driven ego vehicle behaved correctly. That means, the collisions in these cases were caused by the surrounding vehicles and not by ego. These accidents might therefore be irrelevant for the assessment of the ADF. This shows, that the physics-based metric $g^{(\text{RSS Dist})}_{\text{crit.}}$ is not sufficient for the evaluation of an ADF. Hence, the next section discusses the occurrence rate of simulation runs which are critical with regard to the behavior-based metric and which therefore considers faulty behavior of the ADF.

In the discussed optimization procedure, all optimized causality groups respectively parameters were relevant for the critical event set. If some additional parameters, which are irrelevant for the event set, would have been optimized no principal problem would occur. The optimized marginal

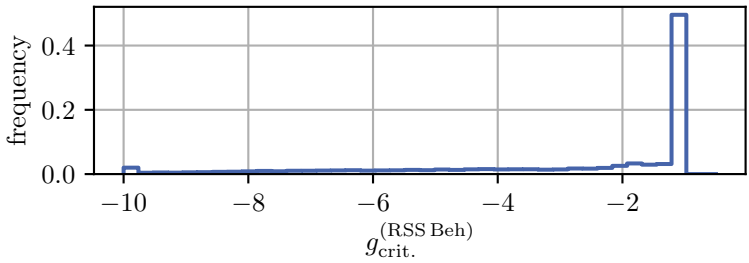


Figure 5.7 The metric values calculated with $g_{\text{crit.}}^{(\text{RSS Beh})}$ for the critical simulation runs drawn from $q\left(\varphi_h \middle| \mathfrak{v}_{\text{RSS Dist}}^*\right)$ in fig. 5.6 which lead to a collision with $g_{\text{crit.}}^{(\text{RSS Dist})} = -1$. About 28 500 critical runs are contained in the histogram. Only in very few runs ego gets a blame for the causation of the collision since the values for $g_{\text{crit.}}^{(\text{RSS Beh})}$ are mostly located at about -1 which means that $r_{\text{frac}}(p, t, \varphi_\xi) l(p, t, \varphi_\xi) \approx 0$ (compare section 4.4.3).

distributions of these irrelevant parameters however would not change and would roughly be distributed as already described by the unoptimized distribution $p_h\left(\varphi_h\right)$.

5.3.4.2 Evaluation on behavior-based metric

The optimization based on $g_{\text{crit.}}^{(\text{RSS Beh})}$ is executed on the parameterization given in table 5.5. The corresponding optimization converges after 7 iterations onto the set

$$\varepsilon_{\text{RSS Beh}} = \left\{\xi \in \mathbf{\Omega} : g_{\text{crit.}}^{(\text{RSS Beh})}\left(\xi, \boldsymbol{\mathcal{S}}\right) \leq -8\right\}. \tag{5.61}$$

Of course, again two additional iterations were performed afterwards in order to ensure a good fit. The resulting parameterization of the parametric family of distributions is denoted by $\mathfrak{v}_{\text{RSS Beh}}^*$. Histograms of the criticalities of simulation runs sampled by $q\left(\varphi_h \middle| \mathfrak{v}_{\text{RSS Dist}}^*\right)$, $q\left(\varphi_h \middle| \mathfrak{v}_{\text{RSS Beh}}^*\right)$ and the original distribution $p_h\left(\varphi_h\right)$ are displayed in fig. 5.8. As expected, $q\left(\varphi_h \middle| \mathfrak{v}_{\text{RSS Beh}}^*\right)$ delivers significantly more samples at behavior values in the range $[-10, -1]$.

The optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}_{\text{RSS Beh}}^*\right)$ is used to approximate the occurrence rate of $\varepsilon_{\text{RSS Beh}}$. The result is presented by fig. 5.9. The optimized distribution determines an occurrence rate of about

$$P_{\text{crit}}\left(\xi \in \varepsilon_{\text{RSS Beh}}\right) \approx 2.79 \times 10^{-6}. \tag{5.62}$$

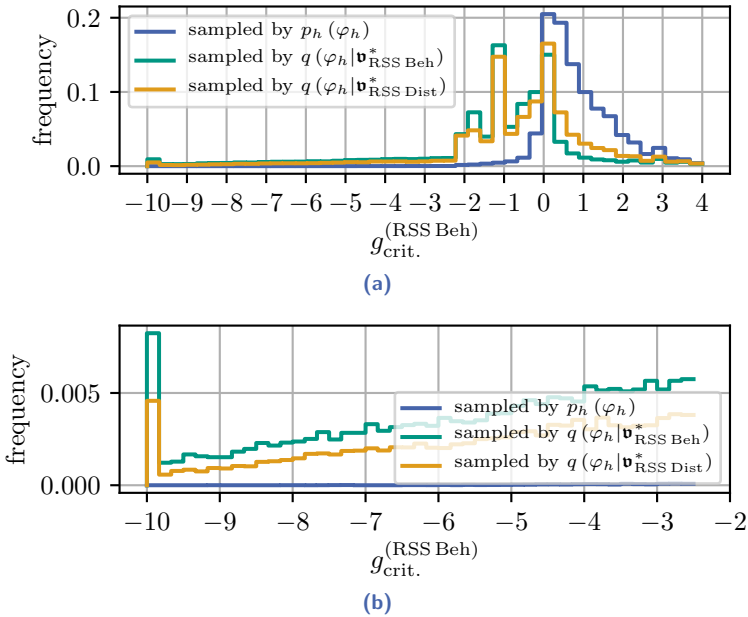


Figure 5.8 Histograms when sampling from the original dataset and the two optimized distributions. The distribution optimized on $g^{(\text{RSS Beh})}_{\text{crit.}}$ increases the share of very critical scenarios which can be blamed on the ego vehicle. **(a)** gives the histograms in the range $[-10, 4]$, whereas **(b)** is limited to $[-10, -2.5]$ in order to give a better overview about the differences at low metric values.

The standard deviation of the occurrence rate curve of the original distribution at sample 400 000 is equal to the optimized distribution's curve at sample 806. This leads to an acceleration factor of

$$\mathfrak{F}^{(\text{RSS Beh})}_{\text{accel.}} \approx 496.3. \tag{5.63}$$

Even after drawing 400 000 samples, the result obtained by the original distribution is not stable. In fig. 5.9, the sampling by $p_h\left(\varphi_h\right)$ only produced 2 critical simulation runs, which then lead to strong fluctuations of the approximated occurrence rate. Hence, a significantly larger amount of simulation runs than the given ones would be necessary to determine a stable approximation by using this original distribution. In order to estimate the needed amount, a comparison to the optimization of $g^{(\text{RSS Dist})}_{\text{crit.}}$ is helpful.

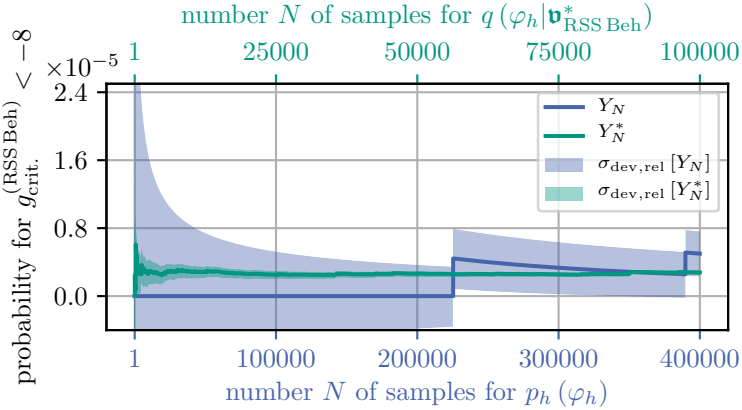


Figure 5.9 Approximation of the occurrence rate of $\varepsilon_{\text{RSS Beh}}$. The approximation is executed both by sampling from the original and the respective optimized distribution. The optimized distribution delivers a stable result. In contrast, the approximation by the original distribution only finds two relevant critical simulation runs and is therefore fluctuating strongly. Many more simulation runs would thus be required to approximate the occurrence rate by the use of the original distribution. This however would exceed the available time budget for the simulations by far.

In this case, a good approximation by the original distribution is achieved after sampling $N^{(\varepsilon_{\text{RSS Dist}})} = 650\,000$ simulation runs. By applying

$$\sigma^{(\varepsilon_{\text{RSS Beh}})}_{\text{dev,rel}} \stackrel{!}{=} \sigma^{(\varepsilon_{\text{RSS Dist}})}_{\text{dev,rel}} \implies \frac{N^{(\varepsilon_{\text{RSS Beh}})}}{N^{(\varepsilon_{\text{RSS Dist}})}} \approx \frac{P_{\text{crit}}\left(\xi \in \varepsilon_{\text{RSS Dist}}\right)}{P_{\text{crit}}\left(\xi \in \varepsilon_{\text{RSS Beh}}\right)}, \quad (5.64)$$

this can be translated to the behavior-based metric. A need of about $N^{(\varepsilon_{\text{RSS Beh}})} = 13\,932\,000$ simulation runs in order to get the same accuracy for the behavior-based metric when sampling by the original distribution is resulting. This high number is practically infeasible to reach since the 650 000 runs already needed 8.4 days[11] to be processed. Consequently, more than 180 days would be required for simulating the 13 932 000 runs. Here, the strength of IS becomes clear: The approximation of $\varepsilon_{\text{RSS Beh}}$ performed by the optimized distribution was feasible and requires much less time.

[11] The simulations were executed with a quad core Intel Xeon E-2144G with 64 GB of RAM.

5.4 Sensitivity analysis

5.4.1 Introduction

In order to realize the described scalable CE optimization approach, it was necessary to identify the parameters $\boldsymbol{\phi}_{\text{rel.}}$ which influence the criticality $g_{\text{crit.}}$ of a simulation run. The relevant parameters used for the previous evaluation were empirically chosen in section 5.3.2 (see table 5.4). This selection of the correct causality groups/relevant parameters is of high importance for the approach to work properly. On the one hand, as few as possible parameters should be chosen in order to keep the amount and the dimensions of the causality groups and the resulting optimizations small and feasible. On the other hand, all parameters which play a role for the criticality of a scenario must be chosen. Otherwise, if some relevant parameters are missed, the optimal importance sampling distribution $p^{*}_{h,\text{opt}}\left(\varphi_h\right)$ in (5.12) cannot be approximated closely enough by the family of functions $q\left(\varphi_h \middle| \varphi_{\mathfrak{v}}\right)$ resulting from (5.19), (5.22), (5.55), (5.56). This might cause severe deviations and losses of efficiency when calculating the occurrence rate of the desired critical event set $\boldsymbol{\varepsilon}$ by applying (5.9). Consequently, a formal method to assess the relevance of the available parameters in $\boldsymbol{\phi}$ with regard to the criticality $g_{\text{crit.}}$ of a scenario is of utmost importance because it allows selecting the causality groups and thereby an appropriate form of the proposed function family.

The assessment of the relevance of the non-physical parameters $\boldsymbol{\phi}$ (see section 5.1) of a simulation run can be grasped as the analysis of the output of a computational model $\mathfrak{M}$ which evaluates to

$$g_{\text{crit.}}^{(\text{expec.})} = \mathfrak{M}\left(\varphi_{\boldsymbol{\phi}}\right). \tag{5.65}$$

That is, the non-physical parameters $\boldsymbol{\phi}$ of the simulation runs are interpreted to be input parameters of $\mathfrak{M}$ and the output is defined by the resulting averaged metric value $g_{\text{crit.}}^{(\text{expec.})}$ of $\mathfrak{N}_{\text{sens.}}$ simulation runs executed during the current evaluation of $\mathfrak{M}$. To fulfill these demands, the computational model $\mathfrak{M}$ is defined by

$$\mathfrak{M}\left(\varphi_{\boldsymbol{\phi}}\right) = \mathbb{E}_{p_h\left(\xi \middle| \boldsymbol{\phi}=\varphi_{\boldsymbol{\phi}}\right)}\left[g_{\text{crit.}}\left(\xi, \boldsymbol{\mathcal{S}}\right)\right], \tag{5.66}$$

which can be approximated by

$$\mathfrak{M}\left(\varphi_{\boldsymbol{\phi}}\right) = \frac{1}{\mathfrak{N}_{\text{sens.}}} \sum_{i=1}^{\mathfrak{N}_{\text{sens.}}} g_{\text{crit.}}\left(\varphi_{\xi_i}, \boldsymbol{\mathcal{S}}\right); \qquad \xi_i \sim p_h\left(\xi \middle| \boldsymbol{\phi} = \varphi_{\boldsymbol{\phi}}\right). \tag{5.67}$$

The challenge to assess the sensitivity of the output $g_{\text{crit.}}^{(\text{expec.})}$ of a computational model $\mathfrak{M}$ with regard to its input values $\varphi_{\boldsymbol{\phi}}$ is addressed by the field of *sensitivity analysis*.

In general, the methods of sensitivity analysis are categorized into the classes of

- *local* methods and
- *global* methods,

as for example discussed by Rivalin et al. [85] and Saltelli et al. [90]. Local methods quantify the influence of a single factor $\mathfrak{p} \in \boldsymbol{\phi}$ while keeping the other factors $\boldsymbol{\phi} \setminus \mathfrak{p}$ constant at a predefined value in the input space. These local methods are not applicable to the presented simulation framework since the simulation models may be highly non-linear and the input parameters might interact strongly with each other.

In contrast, global methods quantify the influence of parameters in the whole input space $\boldsymbol{\Phi}$. However, quantitative global methods usually are very expensive since they require a large number of runs of $\mathfrak{M}$. Consequently, for the purpose of this work, *screening methods* seem to be advantageous. Screening methods evaluate $\mathfrak{M}$'s sensitivity globally. However, they are not capable to derive quantitative results. They can only give qualitative rankings of the model's dependencies on its input parameters. Screening methods are however computationally cheaper and therefore a good choice here since a qualitative sensitivity assessment is sufficient for selecting the relevant parameter set $\boldsymbol{\phi}_{\text{rel.}}$. As Rivalin et al. [85] and Saltelli et al. [90] argue, the *Elementary Effects method* which was developed by Morris [73] in 1991 is the most popular and versatile screening method.

5.4.2 Elementary effects method

For the application of the method of elementary effects [73] on the computational model $\mathfrak{M}$, the input space $\boldsymbol{\Phi} = Val\,(\boldsymbol{\phi})$ of the $\mathfrak{N}_{\text{params}}$-dimensional parameter set $\boldsymbol{\phi}$ is transformed by the bijection $\mathcal{F}_{\text{unit}}$ into a $\mathfrak{N}_{\text{params}}$-dimensional unit hypercube $\boldsymbol{\Phi}^* = \mathcal{F}_{\text{unit}}\,(\boldsymbol{\Phi})$ by normalizing the parameters' ranges. Afterwards, each normalized parameter range is discretized into g-levels $\left\{0, \frac{1}{g-1}, ..., 1\right\}$. Consequently, a $\mathfrak{N}_{\text{params}}$"-dimensional g-level grid"

[19] $\mathbf{\Phi}^*_{\text{grid}}$ of the unit hypercube is obtained. The model $\mathfrak{M}$'s sensitivity with regard to parameter $\mathfrak{p} \in \boldsymbol{\phi}$ is assessed based on its *elementary effects*

$$\mathfrak{d}_{\mathfrak{p}}\left(\varphi_{\boldsymbol{\phi}^*}\right) = \frac{\mathfrak{M}^*\left(\varphi_{\boldsymbol{\phi}^*\setminus\mathfrak{p}^*}, \varphi_{\mathfrak{p}^*} + \Delta_{\text{elementary}}\right) - \mathfrak{M}^*\left(\varphi_{\boldsymbol{\phi}^*\setminus\mathfrak{p}^*}, \varphi_{\mathfrak{p}^*}\right)}{\Delta_{\text{elementary}}}, \tag{5.68}$$

with $\mathfrak{M}^* = \mathfrak{M} \circ \mathcal{F}^{-1}_{\text{unit}}$, $\mathfrak{p}^* = \mathcal{F}_{\text{unit}}(\mathfrak{p})$ and $\boldsymbol{\phi}^* = \mathcal{F}_{\text{unit}}(\boldsymbol{\phi})$. The step $\Delta_{\text{elementary}}$ is a multiple of $\frac{1}{g-1}$. The selection of the evaluation point $\varphi_{\boldsymbol{\phi}^*} \in \mathbf{\Phi}^*_{\text{grid}}$ is subject to $\{\varphi_{\boldsymbol{\phi}^*\setminus\mathfrak{p}^*}, \varphi_{\mathfrak{p}^*} + \Delta_{\text{elementary}}\} \in \mathbf{\Phi}^*_{\text{grid}}$. Since an elementary effect $\mathfrak{d}_{\mathfrak{p}}\left(\varphi_{\boldsymbol{\phi}^*}\right)$ is a local measure of $\mathfrak{p}$'s sensitivity at point $\varphi_{\boldsymbol{\phi}}$, Morris [73] defines a measure

$$\mu = \mathbb{E}\left[\mathfrak{d}_{\mathfrak{p}}\left(\boldsymbol{\phi}^*\right)\right] \tag{5.69}$$

which assesses the sensitivity in a global fashion[12] by approximating the expectation value over randomly drawn $\mathfrak{N}_{\text{morris}}$ evaluation points for each assessed parameter $\mathfrak{p}$. Therefore, $\mathfrak{N}_{\text{morris}} \cdot \mathfrak{N}_{\text{params}}$ evaluation points are required in total. For the sake of efficiency, the $\mathfrak{N}_{\text{morris}}$ values for the different assessed parameters $\mathfrak{p} \in \boldsymbol{\phi}$ are not drawn independently. Instead, Morris suggests drawing the elementary effects in a trajectory-based fashion. An example with 3 sampled trajectories is given in fig. 5.10. Each trajectory has length $\mathfrak{N}_{\text{params}} + 1$ and each parameter changes by $\Delta_{\text{elementary}}$ once in each trajectory. Only one parameter is changed per trajectory step. $\mathfrak{M}^*$ is evaluated at each trajectory point. That allows efficiently calculating an elementary effect $\mathfrak{d}_{\mathfrak{p}}$ for each of the $\mathfrak{N}_{\text{params}}$ parameters during the $\mathfrak{N}_{\text{params}} + 1$ required simulation runs in a trajectory. This is of course accomplished by subtracting the evaluated values of subsequent trajectory points. Since μ is approximated by averaging over $\mathfrak{N}_{\text{morris}}$ evaluations of $\mathfrak{M}$, $\mathfrak{N}_{\text{morris}}$ trajectories thus must be sampled in total. Eventually,

$$\mathfrak{N}_{\text{morris,tot}} = \mathfrak{N}_{\text{morris}} \cdot \left(\mathfrak{N}_{\text{params}} + 1\right) \cdot \mathfrak{N}_{\text{sens.}} \tag{5.70}$$

runs of the ADF's simulation framework must be conducted in order to get an estimation of μ for all examined non-physical parameters. Remember that $\mathfrak{N}_{\text{sens.}}$ was introduced in (5.67).

[12] To be complete, it must be noted that Morris uses a second metric which is based on the standard deviation of the elementary effects. However, since this second metric will not be used in the following no discussion of this metric is given at this point.

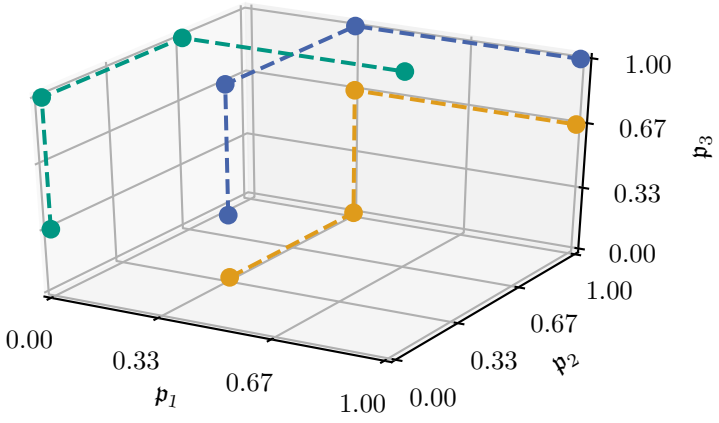


Figure 5.10 Trajectory-based sampling of elementary effects for $\phi = \{\mathfrak{p}_1, \mathfrak{p}_2, \mathfrak{p}_3\}$. Hence, it is $\mathfrak{N}_{\text{params}} = 3$. Here, 3 trajectories are sampled in a $g = 4$ level grid in the 3-dimensional unit hypercube. It is $\Delta_{\text{elementary}} = \frac{2}{3}$.

Morris draws the trajectories randomly. This does not necessarily ensure the trajectories being distributed equally over the whole input grid. If the trajectories are however not distributed as equal as possible, this endangers the expressiveness of the sensitivity analysis since some relevant parts of the input space might not be analyzed as thorough as possible. As a solution, Campolongo et al. [19] propose randomly sampling $\mathfrak{N}_{\text{campo.}}$ trajectories and selecting the set of $\mathfrak{N}_{\text{morris}}$ trajectories whose distances to each other are maximized. This ensures a good input space coverage. Since the calculation of these optimal trajectories becomes very expensive for a large number of parameters and trajectories, Ruano et al. [88] developed a computationally cheaper method which approximately finds the trajectories with the largest distances. This method will be applied in the following.

In addition, Campolongo et al. [19] introduce a revised measure

$$\mu_{\text{campo.}} = \mathbb{E}\left[\left|\mathfrak{d}_{\mathfrak{p}}\left(\phi^*\right)\right|\right]. \tag{5.71}$$

This makes sure, that elementary effects of different sign do not cancel each other. This is especially important for non-linear and interacting input parameters.

Instead of analyzing the elementary effects of single random variables $\mathfrak{p}$, [19] demonstrates, that $\mu_{\text{campo.}}$ can also be used to estimate the model's

dependency on groups of parameters at once. This allows reducing the number of required model runs when analyzing an entire group's importance. This works by moving all factors in a group simultaneously by $\pm\Delta_{\text{elementary}}$ within the trajectory-based sampling. Consequently, the amount of runs of the ADF's simulation framework reduces to

$$\mathfrak{N}_{\text{morris,tot,groups}} = \mathfrak{N}_{\text{morris}} \cdot \left(\mathfrak{N}_{\text{groups}} + 1\right) \cdot \mathfrak{N}_{\text{sens.}}. \tag{5.72}$$

The variable $\mathfrak{N}_{\text{groups}}$ gives the number of groups into which the parameters are separated. Because of the time savings, this group analysis can be applied to identify relevant parameter groups, whose elements then should be analyzed in a more expensive, parameter-wise execution of the screening approach.

5.4.3 Results

When running the sensitivity analysis experiments, $g_{\text{crit.}}^{(\text{RSS Dist})}$ was used for the evaluation in (5.66). The metric was however adapted by setting all resulting criticality values larger 1 to 1. The reason is, that otherwise, because of the diverging nature of the metric, changes in the absolutely safe region of the metric would dominate the results of the analysis (e.g. a change of a metric result from 100 to 150 would cause high elementary effects. They are however not relevant for the assessment, but would still prevail over critical changes like for example changes between -1 and 0.).

When determining the expectation value in (5.66), by drawing $\mathfrak{N}_{\text{sens.}}$ simulation runs, the same $\mathfrak{N}_{\text{sens.}}$ initial scenes $\boldsymbol{x}_0$ are used throughout an entire trajectory. This makes sure, that only the assessed parameters are changed and no other influences bias the analysis' results.

For the simulation framework presented in section 5.3.1, all the parameters are behavior parameters which are given separately for each vehicle in a scenario. Since the distance to the ego vehicle influences the parameters' capability to cause critical scenarios to a great extent, it was chosen to firstly analyze the parameters grouped by their correspondence to a respective surrounding vehicle. The parameterization of this first execution of the elementary effects method is given in the first row of table 5.7. Note that the analysis was limited to show the sensitivity on the nearest 7 vehicles around the ADF controlled ego vehicle which are $\left\{L_{\text{same},0}, L_{\text{same},1}, T_{\text{same},0}, L_{\text{left},0}, T_{\text{left},0}, L_{\text{right},0}, T_{\text{right},0}\right\}$. The results are demonstrated in fig. 5.11(a). Obviously, the most important

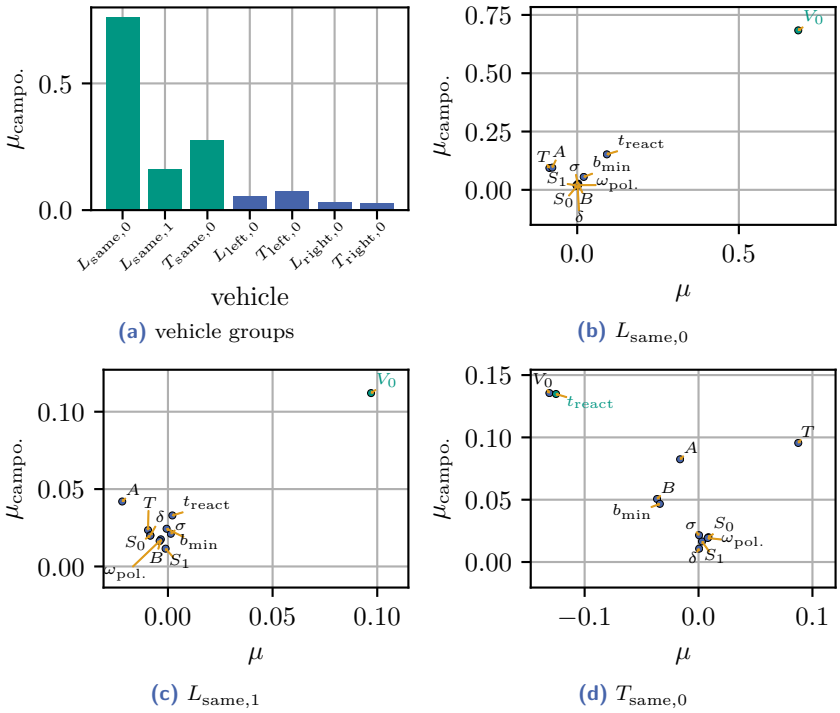
Table 5.7 The elementary effects method was applied twice. The table gives the chosen parameterizations. The last columns shows the amount of runs of the ADF's simulation framework which had to be run for the analyses. Note, that the runs are significantly fewer than the runs required for the unoptimized original determination of the collision rate in section 5.3.4.1.

description	$\mathfrak{N}_{\text{campo.}}$	$\mathfrak{N}_{\text{morris}}$	$\mathfrak{N}_{\text{sens.}}$	$\mathfrak{N}_{\text{groups}}$	g	$\mathfrak{N}_{\text{params}}$	required runs
groups	2000	40	20	7	4	-	6400
single parameters	2000	40	20	-	4	33	27 200

vehicles are $\{L_{\text{same},0}, L_{\text{same},1}, T_{\text{same},0}\}$ which are also the vehicles whose parameters were optimized in section 5.3.4 since they were selected in section 5.3.2. The approach seems to work on the group level since the vehicles which have been found to be relevant in section 5.3.4 also have been selected here.

After having found the most important vehicles with regard to sensitivity, the individual behavior parameters of these vehicles shall now be analyzed in a second sensitivity analysis. The parameterization of this second analysis is also given in table 5.7. The results are displayed by figs. 5.11(b) to 5.11(d). The green marked dots represent the parameters which already were optimized in the IS approach in section 5.3.4 (see table 5.4). Since this IS approach delivered unbiased expectation values, that proves the correctness of the underlying parameter selection. This selection shall therefore be used as a groundtruth to check if the elementary effect methods is able to find all the necessary relevant parameters.

Obviously, for $L_{\text{same},0}$ and $L_{\text{same},1}$ the executed sensitivity analysis is in accordance with the parameters which were optimized for the working approximation of the collision rate in section 5.3.4.1. For the trailing vehicle $T_{\text{same},0}$ the reaction time is also found. However, for $T_{\text{same},0}$ the most important parameter found by the sensitivity analysis is V_0 which was not optimized in the illustrated IS optimization. That can however easily be explained. It is understandable, that the desired velocity V_0 strongly influences the metric result of a simulation run, since low values of the desired velocity induce a slow driving trailing vehicle $T_{\text{same},0}$ and high values cause the vehicle to drive fast. That is, low values cause large distances between ego and $T_{\text{same},0}$. This therefore can reduce the criticality of a simulation run since most of the found accidents in section 5.3.4.1 are



(a) vehicle groups

(b) $L_{\text{same},0}$

(c) $L_{\text{same},1}$

(d) $T_{\text{same},0}$

Figure 5.11 The results for both executions of the elementary effects method. The higher the values of $\mu_{\text{campo.}}$, the more sensitive $\mathfrak{M}$ is to the respective parameter. If a value for μ is negative, then an increase of the respective parameter on average reduces $g_{\text{crit.}}^{(\text{RSS Dist})}$, if the value is positive then the parameter increase also increases the metric. The green marked parameters are the parameters which were selected in section 5.3.2 and which thus were used for the already discussed IS optimizations in section 5.3.4. **(a)** shows the results for the first, group-based execution. **(b)**, **(c)**, **(d)** illustrate the results of the second, parameter-based execution.

caused by the trailing vehicle crashing into ego. In accordance to that, V_0 is located at a negative value of μ, which states that an increase in V_0 reduces $g_{\text{crit.}}^{(\text{RSS Dist})}$ and thus a scenario becomes more critical. A closer look at (5.48) however reveals, that V_0 determines an IDM-controlled vehicle's velocity if it does not follow any leading vehicle. However, V_0 is not dominant anymore when a scene becomes critical and $T_{\text{same},0}$'s distance to ego vanishes; in this case the third term in (5.48) diverges and therefore dominates $T_{\text{same},0}$'s

dynamics. Consequently, V_0 was called to be important by the sensitivity analysis since it can strongly influence a scene's criticality metric value $g_{\text{crit.}}^{(\text{RSS Dist})}$, but it still could be neglected in the CE-optimization since it cannot further decrease $g_{\text{crit.}}^{(\text{RSS Dist})}$ if a scene is already critical.

Consequently, if the CE optimization was executed on the elementary effects method's chosen parameters, an additional, irrelevant parameter would have been included. This however does not pose a problem. In the IS optimization it would become apparent if some parameters are not important since the respective marginal distributions (comp. fig. 5.3) would just not be shifted in contrast to the original distribution. The optimization itself would still be working as long as the sensitivity analysis is not missing any relevant parameter. Consequently, the experiments show that the method of elementary effects is an adequate tool to give a first overview about the parameters' importance. The method therefore enables selecting the relevant causality groups which allows reducing the dimension and number of optimizations which must be executed by the CE method.

5.5 Discussion/summary

In this chapter, a new method to enhance the scalability of CE to complex traffic scenarios is proposed. Previously, IS still was limited to scenarios of limited complexity and class (section 3.3.2). The proposed method mitigates this by focusing the CE optimization on the "relevant" parts of a complete scenario which truly have an influence on a scenario's criticality. That enables a strong reduction of the dimension of the required optimizations and yields contribution 6. The proposed algorithmic scheme was tested and evaluated on an exemplary simulation framework which considers complex highway scenarios whose initial scenes $\boldsymbol{x}_0$ are drawn from the highD dataset. Since all scenes drawn from the dataset can be handled, it is shown that the limitations on complexity and scenario class are strongly relaxed. In this relation, the occurrence rates of two event sets, one defined by a purely physical and the other one defined by a behavior-based metric, were calculated. Hence, contribution 9 is accomplished. The behavior-based metric might produce better insights into an ADF's safety since it filters scenarios whose criticality was caused by the surrounding of the ADF-controlled ego vehicle. The ADF is not to blame for these scenarios.

The proposed methods strongly benefit from knowledge about the "rele-

vant parts" of a scenario. It was shown that the Elementary Effects method detects the "relevant" simulation parameters by judging their influence on an ADF's safety. This addresses contribution 7.

During the simulations, it however became clear, that creating initial scenes by drawing from a dataset is not sufficient, since a pure dataset does not allow optimizing on critical initial scenes. This problem was mitigated by introducing a relaxation phase (see section 5.3.3) which then allowed giving a proof of concept of the whole procedure. However, as already mentioned, the relaxation phase creates unrepresentative/unrealistic simulation runs. Hence, chapter 6 will introduce initial scene models and chapter 7 will show how to combine them with IS in order to get rid of the relaxation phase.

Eventually it must be noted, that the proposed method only works for systems, whose criticality is dominated by a subset of simulation parameters. If all elements of $\boldsymbol{\phi}$ have a similar, high influence on the ADF's safety, no reduction of the CE optimization's dimension could be achieved. However, the likelihood that this case occurs is low since the surrounding vehicles' influence on the ego vehicle reduces when they are farther away or other vehicles are located between them. This decoupling of vehicle behaviors usually should enable a strong reduction of the dimension of the CE optimization when applying the proposed method.

In the future, the approach should be implemented on more complex behavior and ADF models.

6 Scene Generation

As stated before, the development of a scene model, which can be used to generate the initial scenes for the simulation pipeline described in fig. 1.1, is one of the key points of this work. Due to the advantages discussed in section 3.3.1, incremental roadway population models shall be examined and used to realize the scene model. These incremental roadway population models require the modeling of conditional probability distributions (see fig. 3.10). Due to the complexity of the real world, these distributions shall be learned from data. Consequently, efficient and fast methods which enable learning such distributions and allow performing inference on them (determine conditional statements) are essential. In this relation, Bayesian networks (BN) and sum-product networks (SPN) shall be assessed. Therefore, section 6.1 describes the basic methods and concepts needed by BNs and SPNs to learn the probability distributions, to perform inference and to populate a road section. On the way to the realization of the described incremental roadway population model, section 6.2 discusses how to extend the already available approach [119], which only works on straight highway sections, to more complex topologies like intersections (contribution 3). In section 6.3 the use of SPNs instead of BNs (inference on BNs is strained by efficiency problems which SPNs can solve [81, p. 20-21]) is proposed. Additionally, the networks' efficiency and accuracy is compared (contribution 4). Section 6.4 summarizes and concludes the chapter. Parts of this chapter were already published by Jesenski et al. [132, 135];

6.1 Theoretical background

This chapter discusses the methods and concepts needed for the application of BNs and SPNs in order to realize initial scene models. Section 6.1.1 starts by giving detailed insights into the basic concept of the PGM-based sampling framework which realizes the incremental roadway population

model. Section 6.1.2 gives methods for learning and inferencing with BNs and section 6.1.3 depicts the same for SPNs.

6.1.1 Basic idea for scene generation

The principle procedure to scene generation as used within this thesis is conducted by a statistical approach. The latter is based on the work of Wheeler et al. [119] which enables populating straight highway sections. For the purpose of this work, Wheeler's approach was adapted and works as discussed in the following: In principle, the total joint probability distribution of a scene is learned from a discretized dataset. This joint probability distribution contains statistical information about the properties of all vehicles in a scene, and also about their relations. Exemplary properties are velocities, distances between vehicles, directions of travel, widths, lengths and so on. In order to make the reproduction of a scene's joint probability distribution feasible, it is, as already roughly outlined in section 3.3.1.5, factorized into multiple parts on a per-vehicle basis. Afterwards, the scene is reproduced by incrementally sampling lane by lane and vehicle by vehicle. In the following, a simplified view on vehicle properties is utilized: It is approximated that a vehicle is completely defined by $\boldsymbol{\chi} = \{v, x, l, w\}$ which are its velocity, its position on the lane's middle line in the x-direction, its length and its width. In the remainder of this work, the position of a vehicle always is given by the center of its front bumper. For the sake of simplicity, this work does not consider lateral or angular deviations of a vehicle to the middle line of the respective lane.

During sampling, when applying the factorized probability distribution iteratively lane-by-lane, the rightmost lane in the direction of travel (which shall be named lane 1) is by definition populated at first. Afterwards, the next lanes are filled iteratively. The filling procedure within a lane is executed iteratively against the direction of travel in order to make sure that vehicles can be placed accurately in relation to infrastructure elements in front of the vehicles (in contrast to [119], where the lanes are populated in the direction of travel). An example for a 3-lane straight highway section is shown in fig. 6.1. At first, lane 1 is filled by starting at vehicle $(1, 1)$ and iterating to vehicle $(1, n_1)$. Afterwards, vehicles $(2, 1)$ to $(2, n_2)$ and vehicles $(3, 1)$ to $(3, n_3)$ are created. When doing so, the first vehicle in a lane $\mathcal{L}$ is sampled by drawing from

$$v_1, d_\mathrm{b}, l_1, w_1 \sim p_1^{(\mathcal{L})}. \tag{6.1}$$

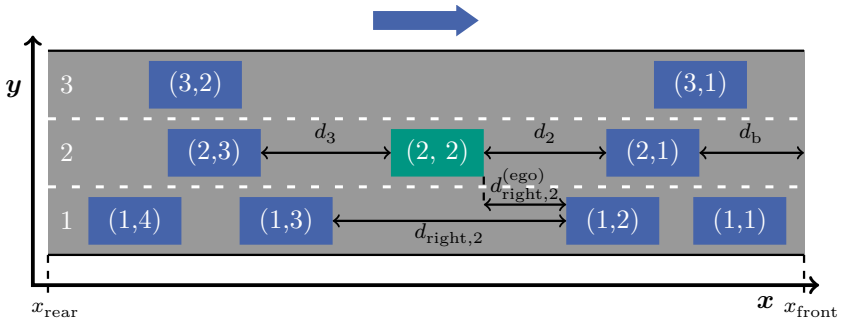


Figure 6.1 Exemplary approach to populate a multi-lane straight highway section. The features relevant for sampling and related to vehicle (2, 2) are visualized. The lane starts at an x-value of x_rear and ends at x_front. The image was already published by Jesenski et al. [136], © 2021 IEEE.

Subsequently, the following vehicles $2, ..., n_\mathcal{L}$ are created by drawing from

$$p^{(\mathcal{L})}_\mathrm{next}\left(\varphi_{v_{i+1}}, \varphi_{d_{i+1}}, \varphi_{l_{i+1}}, \varphi_{w_{i+1}} \middle| \varphi_{v_i}, \varphi_{v_{i-1}}, \varphi_{d_i}, \varphi_{d^{(\mathrm{ego})}_{\mathrm{right},i}}, \varphi_{d_{\mathrm{right},i}}\right). \quad (6.2)$$

Remember, that φ_X denotes a numerically assigned value of random variable X. In the original formulation by Wheeler et al. [119], these distributions were learned by BNs. In section 6.3 the usage of SPNs is proposed.

In (6.1) and (6.2), v_i is the random variable of velocity, l_i the length and w_i the width of the ith vehicle $(\mathcal{L}, i)$ in lane $\mathcal{L}$. Additionally, d_i is the distance between the $(i-1)$th and the ith vehicle. As illustrated in fig. 6.1, d_b gives the distance between the first vehicle in a lane and the boundary x_front of the observed road section. The parameter $d^{(\mathrm{ego})}_{\mathrm{right},i}$ is the distance of the ith vehicle to its predecessor in the lane on its right.[1] The value of $d_{\mathrm{right},i}$ is the distance to the successor of this vehicle (therefore this always describes the distance between i's two nearest surrounding vehicles on the next lane on the right, compare fig. 6.1). The distributions $p^{(\mathcal{L})}_1$ and $p^{(\mathcal{L})}_\mathrm{next}$ must be learned from a (discretized) dataset representing

[1] Note that the random variable's name includes "ego" because the *ith* vehicle is in this step of the sampling procedure understood as the ego vehicle (this is the vehicle whose successor shall be sampled). This enables distinguishing $d_{\mathrm{right},i}$ which denotes a distance of two vehicles in the surrounding of the *current ego i*.

reality. For the sampling of the second vehicle (that means $i = 1$ in (6.2)) in a lane, certain values are not accessible because the predecessor of the predecessor of the vehicle-to-sample is not known, e.g. when sampling $(1, 2)$ no properties of a hypothetical vehicle $(1, 0)$ are known. Hence, let $\varphi_{v_0} = \text{NaN}$ and $\varphi_{d_1} = \text{NaN}$ in this case. These non-existent values are encoded by extra discretization values for these features of the learned probability distribution $p_{\text{next}}^{(\mathcal{L})}$. The same also holds for the features $d_{\text{right},i}^{(\text{ego})}$ and $d_{\text{right},i}$ if the respective predecessor or successor vehicles on the lane to the right of the vehicle-to-sample do not exist.

The features $d_{\text{right},i}^{(\text{ego})}$ and $d_{\text{right},i}$ are necessary to model correlations between different lanes, e.g. if there are a lot of vehicles on lane 1 then the probability that lane 2 is densely populated should also rise. If vehicles on the rightmost lane 1 are sampled, these features are of course neglected. This can be achieved quite easily since the probability distributions $p_1^{(\mathcal{L})}$ and $p_{\text{next}}^{(\mathcal{L})}$ are learned separately for each single lane $\mathcal{L}$. The reason for doing so, is that some properties will fundamentally differ for different lanes. As an example, more trucks will be found on the rightmost lane than on the leftmost lane of a road section. Therefore, the distributions of the lengths l_i in a dataset will be different on each of the lanes. Such deviations in the distributions justify the training of separate networks for the distributions in each lane.

During the placement of the vehicles, the front bumper of the $(i+1)$th vehicle is positioned at

$$x_{i+1} = \begin{cases} x_i - \varphi_{d_{i+1}} - \varphi_{l_i} & \text{for } i \geq 1 \\ x_{\text{front}} - \varphi_{d_{\text{b}}} & \text{for } i = 0 \end{cases}, \tag{6.3}$$

whereas x_i is the x-position of vehicle i on the middle line of the vehicle's lane in the direction of travel (compare fig. 6.1). The sampling within a lane is aborted if $x_{i+1} < x_{\text{rear}}$.

6.1.2 Learning and inference on Bayesian networks

The distributions (6.1) and (6.2) are modeled by using BNs and SPNs. Therefore this section gives an overview on the methods for training and inference which are crucially necessary for BNs.

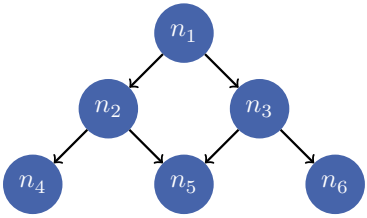


Figure 6.2 Directed acyclic graph $G = (\boldsymbol{V}, \boldsymbol{E})$ of variables $\boldsymbol{V} = \{n_1, n_2, n_3, n_4, n_5, n_6\}$ and edges $\boldsymbol{E} = \{(n_1, n_2), (n_2, n_4), (n_2, n_5), (n_1, n_3), (n_3, n_5), (n_3, n_6)\}$.

6.1.2.1 General properties

Since BNs are probabilistic graphical models, they use a graph structure to encode independencies of a joint probability distribution. This enables the efficient representation of large distributions.

A directed acyclic graph (DAG) $G = (\boldsymbol{V}, \boldsymbol{E})$ used for the construction of Bayesian networks consists of a set of variables $\boldsymbol{V}$ and a set of directed edges $\boldsymbol{E}$ between nodes $n_i, n_j \in \boldsymbol{V}$. A DAG is not allowed to contain directed cycles which means it is not possible to start and end at the same node when following the directions of the edges $\boldsymbol{e} \subset \boldsymbol{E}$ which span a connected path. An example for a DAG G is presented in fig. 6.2.

Formally, a tuple (G, p) consisting of a DAG and a probability distribution p is called Bayesian network, if it fulfills the *Markov condition* [76, Definition 1.9 and p. 40]. If the Markov condition holds, then G expresses a factorization of p [76, Theorem 1.4]. The nodes of G represent random variables and the directed edges $\boldsymbol{E}$ correspond to statistical dependencies. Therefore, in the example given by fig. 6.2, the joint probability distribution can be calculated by

$$\begin{aligned} p(\varphi_{n_1}, \varphi_{n_2}, \varphi_{n_3}, \varphi_{n_4}, \varphi_{n_5}, \varphi_{n_6}) = & p(\varphi_{n_1}) \cdot p(\varphi_{n_2}|\varphi_{n_1}) \cdot p(\varphi_{n_3}|\varphi_{n_1}) \\ & \cdot p(\varphi_{n_4}|\varphi_{n_2}) \cdot p(\varphi_{n_5}|\varphi_{n_2}, \varphi_{n_3}) \\ & \cdot p(\varphi_{n_6}|\varphi_{n_3}). \end{aligned} \tag{6.4}$$

The challenge when working with Bayesian networks is firstly the extraction of the best graph topology describing the statistics of random variables in a dataset (*structure learning*) and secondly the determination of the conditional probability distributions located at each node (*parameter learning*). Additionally, methods for inference, that is the computation of generic

conditional probability distributions of the form

$$p(\varphi_{\boldsymbol{A}}|\varphi_{\boldsymbol{B}}) \text{ with } \boldsymbol{A} \subseteq \boldsymbol{V}, \boldsymbol{B} \subseteq \boldsymbol{V} \setminus \boldsymbol{A}, \tag{6.5}$$

become necessary in order to calculate the distributions described in (6.1) and (6.2).

6.1.2.2 Learning

The following discussion only considers the learning of discretized parameters since this is sufficient for the scope of this work. The implementation of the relevant BN methods was achieved by the application of the python package *pgmpy* [8]. In the following, the used methods are described mathematically.

Parameter learning estimates well fitting conditional probability tables at each of the nodes in $\boldsymbol{V}$. For parameter learning, a Bayesian network is augmented by adding an additional root node $\boldsymbol{f}_i$ at each node n_i [76, Definition 7.3]. For instance, the BN in fig. 6.3 is the augmented version of the BN shown in fig. 6.2. These new nodes allow the introduction of "distributions of distributions". For that, each of the newly introduced nodes contains a set of random variables which consists of

$$\boldsymbol{f}_i = \left\{\boldsymbol{f}_{i,1}, ..., \boldsymbol{f}_{i,q_i}\right\}. \tag{6.6}$$

The entirety of parents of node n_i can take q_i different instantiations. Each of the elements of the set $\boldsymbol{f}_i$ is a further set containing

$$\boldsymbol{f}_{i,j} = \left\{f_{i,j,1}, ..., f_{i,j,r_i}\right\}. \tag{6.7}$$

The size of r_i gives the number of discrete bins for variable i described by node n_i. A conditional probability distribution shall be defined by

$$p\left(\varphi_{n_i}\middle|j, \varphi_{\boldsymbol{f}_i}\right) = \varphi_{f_{i,j,\varphi_{n_i}}} \tag{6.8}$$

That is, $\varphi_{f_{i,j,\varphi_{n_i}}}$ gives the probability for the occurrence of φ_{n_i} given the parental instantiation j. Consequently, the distribution of random variable n_i conditioned on the parental instantiation j is completely described by the set $\boldsymbol{f}_{i,j}$. As a result, the distribution

$$p_{\boldsymbol{f}_{i,j}}\left(\varphi_{f_{i,j,1}}, ..., \varphi_{f_{i,j,r_i}}\right) \text{ with } \varphi_{f_{i,j,k}} \in [0,1], \tag{6.9}$$

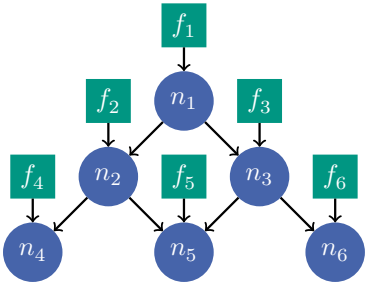


Figure 6.3 Augmented Bayesian network. The inherent embedded Bayesian network is the network shown in fig. 6.2. The newly introduced nodes marked by the squared, green boxes are new root parents of the original variables in $\boldsymbol{V}$. They result in the description of distributions of distributions and thereby allow performing Bayesian learning.

and $\varphi_{f_{i,j,r_i}} = 1 - \sum_{k=1}^{r_i - 1} \varphi_{f_{i,j,k}}$ describes a distribution of possible distributions $p\left(\varphi_{n_i} \middle| j, \varphi_{\boldsymbol{f}_i}\right)$ [76, p. 392]. Note that this includes independency assumptions

$$p_{\boldsymbol{f}_i}\left(\varphi_{\boldsymbol{f}_i}\right) = \prod_{j=1}^{q_i} p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}}\right). \tag{6.10}$$

The aim of Bayesian parameter learning is to find posteriors

$$p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}} \middle| \boldsymbol{d}\right) = \frac{p_{\boldsymbol{f}_{i,j}}\left(\boldsymbol{d} \middle| \varphi_{\boldsymbol{f}_{i,j}}\right) p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}}\right)}{\int_{Val(\boldsymbol{f}_{i,j})} p_{\boldsymbol{f}_{i,j}}\left(\boldsymbol{d} \middle| \varphi_{\boldsymbol{f}_{i,j}}\right) p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}}\right) \mathrm{d}\varphi_{\boldsymbol{f}_{i,j}}}. \tag{6.11}$$

The element $\boldsymbol{d}$ describes a training dataset [76, Definition 7.4] of type

$$\boldsymbol{d} = \{d_1, ..., d_M\} \tag{6.12}$$

with length M. The data tuples $d_m \in \boldsymbol{d}$ have the shape of

$$d_m = \begin{pmatrix} \varphi_{n_1}^{(m)} \\ \vdots \\ \varphi_{n_N}^{(m)} \end{pmatrix}. \tag{6.13}$$

The parameter $N = |\boldsymbol{V}|$ gives the total number of random variables included in the BN. Consequently, it is

$$p_{\boldsymbol{f}_{i,j}}\left(\boldsymbol{d}\middle|\varphi_{\boldsymbol{f}_{i,j}}\right) = \prod_{k=1}^{r_i} \varphi_{f_{i,j,k}}^{s_{i,j,k}}. \tag{6.14}$$

The value of $s_{i,j,k}$ specifies the number of data tuples in $\boldsymbol{d}$ for which the ith feature takes the kth value and for which the parent configuration of feature i is in its jth instantiation [76, Theorem 7.7].

The usual choice for the a priori distribution $p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}}\right)$ is the Dirichlet distribution

$$\mathrm{Dir}\left(\varphi_{\boldsymbol{f}_{i,j}}; \boldsymbol{a}_{i,j}\right) = \frac{\Gamma\left(\sum_{k=1}^{r_i} a_{i,j,k}\right)}{\prod_{k=1}^{r_i} \Gamma\left(a_{i,j,k}\right)} \prod_{k=1}^{r_i} \varphi_{f_{i,j,k}}^{a_{i,j,k}-1} \tag{6.15}$$

with $\varphi_{f_{i,j,k}} \in [0,1]\,; \sum_{k=1}^{r_i} \varphi_{f_{i,j,k}} = 1$. It is $\boldsymbol{a}_{i,k} = \left\{a_{i,j,k}\middle|1 \leq k \leq r_i\right\}$ It can be shown, that the Dirichlet distribution is a *conjugate prior* with respect to the the categorical likelihood function in (6.14) which means, that the posterior distribution remains in the same family of functions. Therefore, it is (see [76, Corollary 7.7])

$$p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}}\middle|\boldsymbol{d}\right) = \mathrm{Dir}\left(\varphi_{\boldsymbol{f}_{i,j}}; \boldsymbol{a}_{i,j} \oplus \boldsymbol{s}_{i,j}\right). \tag{6.16}$$

This follows from the application of the Dirichlet-Distribution into (6.11). $\oplus$ is the element-wise addition of two sets. Let $\boldsymbol{s}_{i,j} = \left\{s_{i,j,k}\middle|1 \leq k \leq r_i\right\}$. The parameter learning of the trained embedded Bayesian network then results in

$$\begin{aligned} p\left(\varphi_{n_i}\middle|j,\boldsymbol{d}\right) &= \int\limits_{Val\left(\boldsymbol{f}_{i,j}\right)} p\left(\varphi_{n_i}\middle|j,\varphi_{\boldsymbol{f}_{i,j}}\right) p_{\boldsymbol{f}_{i,j}}\left(\varphi_{\boldsymbol{f}_{i,j}}\middle|\boldsymbol{d}\right) \mathrm{d}\varphi_{\boldsymbol{f}_{i,j}} \\ &= \sum_{k=1}^{r_i} \delta_{n_i,k} \mathbb{E}_{p_{\boldsymbol{f}_{i,j}}\left(\boldsymbol{f}_{i,j}|\boldsymbol{d}\right)}\left[f_{i,j,k}\right]. \end{aligned} \tag{6.17}$$

As discussed by Neapolitan [76, Corollary 7.5, Corollary 7.4], when using Dirichlet-Distributions it follows that

$$p\left(\varphi_{n_i}\middle|j,\boldsymbol{d}\right) = \frac{a_{i,j,\varphi_{n_i}} + s_{i,j,\varphi_{n_i}}}{\sum_{k=1}^{r_i} a_{i,j,k} + s_{i,j,k}}. \tag{6.18}$$

In order to obtain this result, it is necessary to specify the prior by choosing the values in the parameter set $\boldsymbol{a}_{i,j}$. Since no prior knowledge shall influence the training of the BN, a uniform prior distribution with

$$a_{i,j,k} = \frac{N}{r_i q_i} \tag{6.19}$$

is chosen [76, Theorem 7.8]. N is called *equivalent sample size*.

Structure learning of BNs was performed by using a *greedy search algorithm*. Hence, a metric which quantifies the quality of fit of a DAG G is required. This quantification can be performed by using

$$p(G|\boldsymbol{d}) = \frac{p(\boldsymbol{d}|G)p(G)}{p(\boldsymbol{d})}. \tag{6.20}$$

Uniform priors which encode missing prior knowledge will be used. Hence, the quantification reduces to the evaluation of

$$p(\boldsymbol{d}|G) = \int p(\boldsymbol{d}|\varphi_{\boldsymbol{f}}, G) \cdot p(\varphi_{\boldsymbol{f}}|G)\, \mathrm{d}\varphi_{\boldsymbol{f}}. \tag{6.21}$$

Neapolitan [76, Definition 8.2] shows, that when a Dirichlet distribution is used for the prior, this expression is given by

$$p(\boldsymbol{d}|G) = \prod_{i_1}^{n} \prod_{j=1}^{q_i^{(G)}} \frac{\Gamma\left(\sum_k^{r_i} a_{i,j,k}^{(G)}\right)}{\Gamma\left(\sum_k^{r_i} a_{i,j,k}^{(G)} + s_{i,j,k}^{(G)}\right)} \prod_{k=1}^{r_i} \frac{\Gamma\left(a_{i,j,k}^{(G)} + s_{i,j,k}^{(G)}\right)}{\Gamma\left(a_{i,j,k}^{(G)}\right)}. \tag{6.22}$$

This metric is denoted as *Bayesian scoring criterion*. For the selection of the priors again the condition in (6.19) is applied. Note that the superscript (G) shows that the relevant values are dependent on the structure of the DAG.

Ideally, it would be best to use this metric to test and assess all possible BN structures. However, Neapolitan [76, p. 450-451] illustrates, that the number of possible structures strongly increases with network size $|\boldsymbol{V}|$ and it quickly becomes infeasible to exhaustively test them all. Consequently, approximate learning algorithms which heuristically search the space of possible structures are utilized. In this thesis, a greedy structure search algorithm [76, Algorithm 9.2] which iteratively manipulates the structure of the network by using the single-edge operations of

1. adding an edge,
2. removing an edge and
3. reversing an edge,

in order to maximize the Bayesian scoring criterion, is used. The algorithm stops when finding a local maximum. In order to limit the complexity and time requirements of the structure learning algorithm, the number of allowed parents per node shall be limited to three. Additionally, the search algorithm is not allowed to reverse modifications applied during the last two iterations.

6.1.2.3 Inference

After the learning of network parameters and structure is accomplished, an algorithm for inference is required for using the network to obtain information about conditional relationships between variable sets as formulated in (6.5). Marginalization is sufficient for performing inference since

$$p\left(\varphi_{\boldsymbol{V}_1}|\varphi_{\boldsymbol{V}_2}\right) = \frac{p\left(\varphi_{\boldsymbol{V}_1 \cup \boldsymbol{V}_2}\right)}{p\left(\varphi_{\boldsymbol{V}_2}\right)} = \frac{\sum_{\boldsymbol{V}\setminus(\boldsymbol{V}_1\cup\boldsymbol{V}_2)} p\left(\varphi_{\boldsymbol{V}}\right)}{\sum_{\boldsymbol{V}\setminus\boldsymbol{V}_2} p\left(\varphi_{\boldsymbol{V}}\right)}, \tag{6.23}$$

for $\boldsymbol{V}_1 \cup \boldsymbol{V}_2 \subset \boldsymbol{V}$ and $\boldsymbol{V}_1 \cap \boldsymbol{V}_2 = \emptyset$, when the joint p is defined over the random variables in $\boldsymbol{V}$. Because marginalization requires summing out all parameter combinations, the effort to calculate $p\left(\varphi_{\boldsymbol{V}_1}|\varphi_{\boldsymbol{V}_2}\right)$ completely for all variable values scales by $\mathcal{O}\left(r_{\max}^{|\boldsymbol{V}|}\right)$, where $r_{\max} = \max\{r_i | i \in \boldsymbol{V}\}$ is the maximal number of discrete bins of all variables. Since this growth quickly makes marginalization infeasible when the network size $|\boldsymbol{V}|$ grows, approaches to enable efficiency gains were developed.

Here, the approach of *Variable Elimination* [56, chapter 9] shall be applied. The used property is that the joint probability distribution in a BN is always given by a set of factors, whereas the scope of each factor will not contain all random variables in $\boldsymbol{V}$. This allows calculating the marginalization of some variables more efficiently by shifting some parts of the sum "backwards" in the product of probabilities and by caching factors ϕ_X which correspond to the marginalization of variable X. For instance, let's assume a joint probability function with $\boldsymbol{V} = \{a, b, c\}$ which as an example might be given by

$$p\left(\varphi_a, \varphi_b, \varphi_c\right) = p\left(\varphi_a\right) \cdot p\left(\varphi_b|\varphi_a\right) \cdot p\left(\varphi_c|\varphi_b\right). \tag{6.24}$$

The marginalization on b can then be expressed by

$$\begin{aligned} p(\varphi_b) &= \sum_{a,c} p(\varphi_a)p(\varphi_b|\varphi_a)p(\varphi_c|\varphi_b) \\ &= \underbrace{\sum_{a} p(\varphi_a) \cdot p(\varphi_b|\varphi_a) \cdot \underbrace{\sum_{c} p(\varphi_c|\varphi_b)}_{\phi_c(\varphi_b)}}_{\phi_a(\varphi_b)} . \end{aligned} \tag{6.25}$$

Remember, that r_X gives the number of discrete values available for random variable X. When iteratively calculating and caching $\phi_c(\varphi_b)$ and $\phi_a(\varphi_b)$, $r_b \cdot (r_c - 1)$ summation operations must be conducted for $\phi_c(\varphi_b)$ and $r_b \cdot (r_a - 1)$ summations and $2 \cdot r_a \cdot r_b$ multiplication operations for $\phi_a(\varphi_b)$. That means, the number of necessary arithmetic operations is strongly dependent on the size of the factors before marginalization (in this case $r^{(\text{fac})}_{\phi_a} = r_a \cdot r_b$ and $r^{(\text{fac})}_{\phi_b} = r_b \cdot r_c$). Further analysis shows that the number of operations needed by the variable elimination approach can in general be approximated by $\mathcal{O}\left(|\boldsymbol{V}| \cdot r^{(\text{fac})}_{\max}\right)$ [56, p. 306]. The parameter $r^{(\text{fac})}_{\max}$ gives the size of the largest factor before marginalization (here: $\max\{r^{(\text{fac})}_{\phi_a}, r^{(\text{fac})}_{\phi_b}\}$). As shown, the sizes of the factors scale exponentially with the factors' scopes (here the scopes are: $\{a, b\}$ and $\{b, c\}$). Hence, if a marginalization by shifting the sums "backwards" creates factors with small scopes, variable elimination can strongly speed up the calculation of marginals and thus of conditional distributions since the number of operations scales exponentially with the scope of the factors and not with the total scope $|\boldsymbol{V}|$ of the total network. Since different orders of summing out the variables will create factors of different sizes, the search for good *elimination orderings* with small factors is crucial. In this work, pgmpy's implementation of the *Min-fill* algorithm [56, chapter 9.4.3.2] is used. Note however, that depending on the structure of the BN, large factors may arise. Therefore, this algorithm will not work properly on all possible networks. Hence, inference may still need time exponential to $|\boldsymbol{V}|$ in the worst case.

6.1.3 Learning and inference on sum-product networks

Since inference may still scale badly with the size of a BN, sum-product networks may be an advantageous alternative. This section discusses the necessary methods to train and use them.

6.1.3.1 General properties

In 2011 Poon and Domingos [83] have proposed a new type of deep network which they called sum-product networks (SPN). This type of network shall overcome the traditionally existent challenge of efficiency when performing exact inference on classic probabilistic models such as BNs. An elegant, recursive method to define SPNs is given by Gens and Domingos [32].

Definition 6.1 (sum-product networks [32])
A sum-product network (SPN) is defined as follows.

1. A tractable univariate distribution is an SPN.
2. A product of SPNs with disjoint scopes is an SPN.
3. A weighted sum of SPNs with the same scope is an SPN, provided all weights are positive.
4. Nothing else is an SPN.

The second requirement is called *decomposability*. The third one is denoted as *completeness*. The *scope* of an SPN is the set of variables on which it is defined.

Structures fulfilling the requirements of definition 6.1 can be represented by the graph type illustrated in fig. 6.4. Similar to the graphs representing BNs, the SPN graphs are *directed* and *acyclic*. In addition, the graphs contain only one *root* per modeled probability distribution. In contrast to BNs, the SPN graphs' nodes do not contain random variables but the operations requested by requirements 2 and 3 in the definition. Additionally, the edges do not correspond to statistical dependencies but connect the input and output elements of the operations. An SPN is evaluated from the bottom to the top. That means, in the beginning the univariate probability distributions p_u in the SPN's leaves are evaluated at the chosen values of the random variables. Afterwards, the sum and product nodes are applied. Sum nodes contain the weighted sums of their child nodes, whereas product nodes contain the products of their children. The joint (unnormalized) probability distribution $\mathcal{N}(\varphi_{\boldsymbol{x}})$ which shall be modeled by the SPN is then

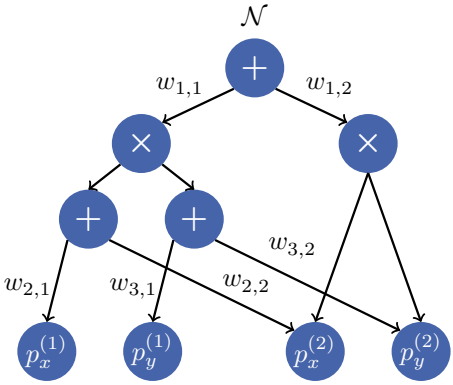


Figure 6.4 Example for the graph-based representation of an SPN. The SPN represented by this particular graph models a bivariate probability distribution $\mathcal{N}(\varphi_x, \varphi_y)$. In contrast to the type of graphs which defines BNs, the graph does not describe random variables in its nodes, but operations applied to the child nodes. As given in definition 6.1, these operations are sums represented by ⊕ and products represented by ⊗. p_u gives the univariate distributions in the leaf nodes. The parameters $w_{h,i}$ give the weights of the sums. The image is based on fig. 2 published by Jesenski et al. [135], © 2020 IEEE.

given by the root node. The related normalized probability distribution is obtained by

$$p\left(\varphi_{\boldsymbol{x}}\right) = \frac{\mathcal{N}(\varphi_{\boldsymbol{x}})}{Z}. \tag{6.26}$$

$Z = \int_{Val(\boldsymbol{x})} \mathcal{N}(\varphi_{\boldsymbol{x}})\, \mathrm{d}\varphi_{\boldsymbol{x}}$ denotes the partition function of the distribution. If the SPN is normalized by $\sum_i w_{h,i} = 1, \forall h \in \mathfrak{P} \subseteq \boldsymbol{V}$ and the leaf distributions are normalized, then it is $Z = 1$ [32, p.2]. The set $\mathfrak{P}$ contains all sum nodes of the SPN. The set $\boldsymbol{V}$ again contains all nodes of the SPN.

Since a marginalization over $\mathcal{N}(\varphi_{\boldsymbol{x}})$ can be realized by directly marginalizing over the univariate distributions in an SPN's leave nodes, inference scales only linearly with the number of edges in the network [32]. Therefore, SPNs allow the efficient calculation of conditional probability values as demanded in (6.1) and (6.2). Similar to BNs, the challenges of applying SPNs are to learn a good structure and parameters (for SPNs that are the weights in the sums) in order to create SPNs which fit properly onto real statistical probability distributions. The algorithms used to realize this learning procedure are described in the next section.

6.1.3.2 Learning

There are lots of different SPN learning methods available in the literature. For the purpose of this work, *LearnSPN* (LSPN) [32], *Mixed SPN* (MSPN) [71], *SearchSPN* (SSPN) [24] and *Online SearchSPN* (OSSPN) [25] were tested and compared to each other for the use case of populating a road topology. These four algorithms were chosen by considering their specific strengths and weaknesses. The python library *SPFlow* [72] was used for training and inference on the SPNs.

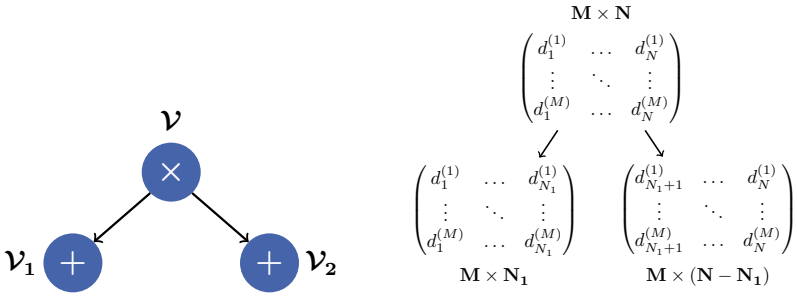
A dataset whose structure shall be learned by an SPN can be described as $M \times N$ data matrix

$$d_{\boldsymbol{\mathcal{V}}} = \overset{\text{variables}}{\rightarrow} \begin{pmatrix} d_1^{(1)} & \dots & d_N^{(1)} \\ \vdots & \ddots & \vdots \\ d_1^{(M)} & \dots & d_N^{(M)} \end{pmatrix} \overset{\text{instances}}{\downarrow} \tag{6.27}$$

with $|\boldsymbol{\mathcal{V}}| = N$ random *variables* in the horizontal direction and M *instances* of variable tuples in the vertical direction. $d_i^{(j)}$ gives the jth repetition of the value of the ith random variable. In principle, the examined learning algorithms work by the definition of frameworks which allow splitting these instances and variables into subsets.

LSPN (LearnSPN) is one of the earliest and most popular structure (and parameter) learning algorithms for SPNs. Its developers call it the "first algorithm for learning the structure of SPNs that does not sacrifice any of their expressiveness" [32]. In fact, LSPN is an algorithmic scheme and no fully defined algorithm itself, since it describes a top-down approach to split a dataset into subsets of *instances* and *variables*. This splitting however may be executed using various algorithms. The algorithmic scheme consists of three types of basic steps, which are

1. fit of univariate distributions,
2. instance splits, and
3. variable splits.



(a) Exemplary product node. The child nodes are not necessarily sum nodes.

(b) Splitting of a dataset at a product node corresponding to the situation in **(a)**.

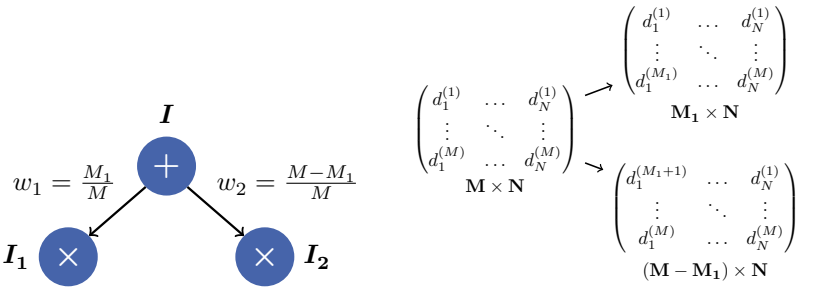
Figure 6.5 Example for binary variable splitting. **(a)** illustrates the creation of a product node and **(b)** displays the respective dataset splitting. $\boldsymbol{\mathcal{V}}$ describes the scope of the parent node and $\boldsymbol{\mathcal{V}_1}, \boldsymbol{\mathcal{V}_2}$ give the scopes of the child nodes. $|\boldsymbol{\mathcal{V}_1}| = N_1$ and $|\boldsymbol{\mathcal{V}_2}| = N - N_1$ are the sizes of the independent variable subsets.

Since LSPN works top-down and starts with the root node, the created SPNs possess a tree-like shape.

A fit of univariate distributions is performed when a dataset containing only one random variable is left. Besides, for a dataset with multiple variables this is applied if too few instances are left in the dataset for the other two basic steps to be performed. In this case, the dataset is naively factorized into univariate distributions. This basic step is equivalent to the creation of univariate leaf nodes p_u.

Variable splits are applied if an independency measure applied onto the dataset is able to find subsets of random variables which are independent to each other. If such independent subsets are successfully found, the dataset is split into data subsets which correspond to these independent variable sets. Afterwards, the LSPN algorithm recurses onto these newly found data subsets. This step is equivalent to the creation of a product node $\times$. The splitting procedure is illustrated in fig. 6.5.

Instance splits are applied if the other two cases are not possible. That means, the dataset contains multiple random variables and there are



(a) Exemplary sum node. The child nodes are not necessarily product nodes.

(b) Splitting of a dataset at a sum node corresponding to the situation in **(a)**.

Figure 6.6 Example for instance splitting into two subsets. **(a)** shows the creation of a sum node. In **(b)** the respective dataset splitting is discussed. $\boldsymbol{I}$ describes the M instances of the parent node and $\boldsymbol{I_1}, \boldsymbol{I_2}$ identify the instances of the child nodes. $|\boldsymbol{I_1}| = M_1$ and $|\boldsymbol{I_2}| = M - M_1$ are the sizes of the instance subsets.

no independent variable subsets. In this case, the algorithm tries to find subsets of instances (data samples) which are "different" to each other and do not follow similar distributions. This can be achieved by applying clustering algorithms (k-means, ...). Having clustered such instance subsets, the LSPN algorithm is applied recursively to each of them. This basic step corresponds to the creation of a sum node ⊕ in an SPN and is equivalent to defining a probability mixture distribution. The weights of the sum nodes are given by the relative sizes of the newly found instance subsets. The instance splitting procedure is illustrated in fig. 6.6.

MSPN (Mixed SPN) is respectively an augmentation or instantiation of LSPN. That means, MSPN uses the very same algorithmic scheme. One of the disadvantages of LSPN in its original formulation is, that it is required to choose a certain parametric family of functions [71]. The usually used methods for variable and instance splitting depend on the parametric form. The same is of course valid for the fitting of the univariate distributions. MSPN mitigates this by being able to train SPNs without specifying a concrete parametric family of distributions. Additionally, MSPN is able to work on mixed domains of discrete and continuous data. The key idea of MSPN is to evaluate the variables and instances' non-linear statistical dependency in a k-dimensional space which is created by applying a pipeline

of non-linear transformations onto the original data. This pipeline is based onto the randomized dependency coefficient (RDC) [68]. For the univariate leaf distributions, the algorithm utilizes histogram leaves with Laplacian smoothing [71].

SSPN (SearchSPN) is an alternative learning framework for SPNs. In contrast to LSPN and MSPN, it is however not limited to the learning of tree-shaped SPNs. As the basic difference to the latter two approaches, SSPN is not a top-down algorithm which learns an SPN starting at the root node and ending at the leaf nodes. Instead SSPN uses an initial seed model and augments it by selecting the product node whose surrounding reduces an approximation of the corresponding dataset likelihood the most. This approximation of the likelihood is defined by the use of *maximal complete sub-circuits* extracted for each instance of training data [25]. By using the parameterization of these maximal complete sub-circuits, the training dataset then can be transformed in *mini-datasets*. The creation of these mini-datasets allows the factorization of the approximated likelihood into units associated with single product nodes. Hence, it is possible to separately evaluate the influence of the product nodes on the total approximate likelihood.

The product node which corresponds to the worst-performing likelihood unit is then selected and the structure in its surrounding is updated in order to fit better onto the dataset. These structural changes are accomplished by firstly checking the independency assumptions between the product node's child nodes (independency of the variables in the scopes of the childs) which can be accomplished by again using the mini dataset and a dependency measure. After independent child subsets have been found, the fitting within these internally dependent subsets is improved by cloning the subsets' nodes and by incorporating an additional layer of sum and product nodes for each subset [25]. This corresponds to the introduction of a probability mixture model in order to grasp the subset-internal dependencies. Figure 6.7 displays an example for this introduction of additional nodes into an already existent SPN.

The iterative selection of the worst product node and consequently the improvement of the surroundings of these product nodes, results in the construction of an SPN which fits onto the training dataset. Within the scope of this work the modified version of SSPN as given in [25] is utilized. For more details on the implementation see [24, 25].

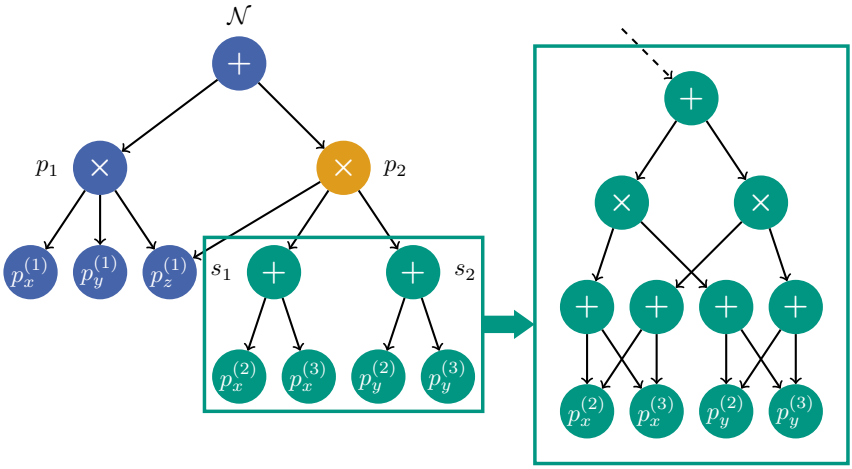


Figure 6.7 Illustration of SSPN-based improvement of the structure in the surrounding of a selected product node. In this example, node p_2 is the product node with the worst likelihood unit. When checking the independency assumptions of the childs, the independent subsets $V_1 = \left\{p_z^{(1)}\right\}$ and $V_2 = \{s_1, s_2\}$ were found. Since $|V_1| = 1$ only the structure of V_2 is adapted. Therefore, new nodes constructing a mixture are integrated into the initial SPN. Compare also [25, Fig. 2].

OSSPN (Online SearchSPN) is the fourth tested algorithm. Since SSPN is not a top-down approach, but iteratively augments an already existing SPN at its worst location, it is predestined to be used in an online setting. The benefit of an online approach for the generation of scenes is clear: Firstly, an existing model could continuously be improved and changed when traffic data are measured for a longer time period and when the world is changing during this time. Secondly, training becomes possible on potentially very large datasets which do not fit into the memory of the system used for the training. For these reasons, the approach proposed by Dennis and Ventura [25] which is an adaption of SSPN onto the online setting is examined. In the original formulation of OSSPN, parameter learning was operated on the last r data tuples, whereas the structure learning step was performed on all batches until the current one. In the remainder of this work, this is changed to the last r parameter tuples for learning both parameters and structure in order to make sure, that the second benefit of not having to load all data into memory holds.

6.2 Generalization of the model to intersections

This section proposes a framework which can be used to extend the applicability of the incremental roadway population model (originally introduced by Wheeler et al. [119] and adapted in section 6.1.1) to more complex road topologies like e.g. intersections. For that, section 6.2.1 gives the challenges which must be solved and section 6.2.2 proposes a model to solve these mentioned challenges. In section 6.2.3 the preprocessing of the datasets used to train the model is described and in section 6.2.4 the trained model is evaluated. For the purpose of this section, BNs shall be used to represent the stochastic distributions in the model. This section was already partly published by Jesenski et al. [132]; © 2019 IEEE.

6.2.1 Problem statement

In this section, the incremental roadway population model proposed in [119] shall be extended to intersections. In order to do so, the following challenges concerning complex road topologies must be solved:

1. *Inhomogeneous lanes:* In contrast to the straight highway sections handled in [119] and illustrated in fig. 6.1, a lane $\mathcal{L}$ with length $s_2 - s_1$ in an intersection/complex road topology will be represented by a parameterized curve of type

$$\boldsymbol{f}_{\mathcal{L}} : [s_1, s_2] \to \mathbb{R}^2. \tag{6.28}$$

   The curve's parameter $s \in [s_1, s_2]$ gives the arc length within the lane. Since lanes are inhomogeneous, the position of a vehicle on these lanes will have strong implications on its behavior and properties. Think for example about the position in front of a stopping line. A vehicle which is located directly in front of the stopping line is more likely to drive slowly than a vehicle with a large distance to the stopping position. Therefore, the conditional distribution (6.2) must definitively be adapted to contain vehicle arc-length positions. This introduces the necessity of being able to handle positions on such arbitrary parameterized curves. Since especially longitudinal and lateral relations to the lane are relevant, global coordinates (x, y) are not convenient. Hence, a Frenet frame based on the lanes' middle lines will be introduced in section 6.2.2.1.

2. *Merging, splitting & intersecting lanes:* In an intersection, lanes can merge, split or intersect. Thus, a model used to generate scenes has to take into account the interactions between vehicles located in such related lanes. As a solution, the detachment of lanes into lane sections

$$\boldsymbol{f}_{\mathcal{L}}(t) = \begin{cases} \boldsymbol{f}_{\mathcal{L}_1} & \text{for } s \in [s_1, s_{11}] \\ \boldsymbol{f}_{\mathcal{L}_2} & \text{for } s \in (s_{11}, s_{12}] \\ \vdots & \\ \boldsymbol{f}_{\mathcal{L}_n} & \text{for } s \in (s_{1n}, s_2] \end{cases} \tag{6.29}$$

is proposed in section 6.2.2.2. In line with this, a scheme to populate the detached lane sections $\{\mathcal{L}_1, ..., \mathcal{L}_n\}$ based on boundary conditions is developed in section 6.2.2.4.

3. *Interactions between lanes and global parameters:* When considering complex intersections, there will be global parameters affecting all lane sections. Examples are parameters which determine technical properties such as traffic light phase or environmental properties like temperature or clock time. These global parameters necessitate a hierarchization of parameters in the sampling process. Section 6.2.2.3 therefore suggests the use of *intersectional* and *intrasectional* parameters whose sampling is realized separately by different networks.

6.2.2 Model description

This section now proposes a new generic model which allows handling the mentioned challenges. The application of this model is depicted at the example of an intersection in Aschaffenburg contained in the Ko-PER dataset (appendix B.1) and an intersection in Aachen which is part of the inD dataset (appendix B.3).

6.2.2.1 Frenet frame

To solve the challenge of *inhomogeneity* of lanes, a 2-dimensional Frenet frame is used. This means a point on a lane $\mathcal{L}$ is defined by its position relative to the lane's centerline $\boldsymbol{f}_{\mathcal{L}}$. For that, the transformation

$$\mathcal{F}_{\boldsymbol{f}_{\mathcal{L}}} : \begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} s \\ \rho \end{pmatrix} \tag{6.30}$$

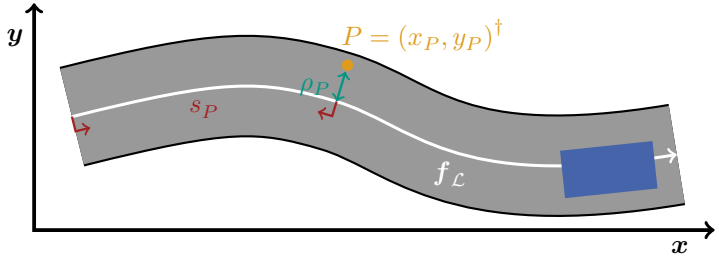
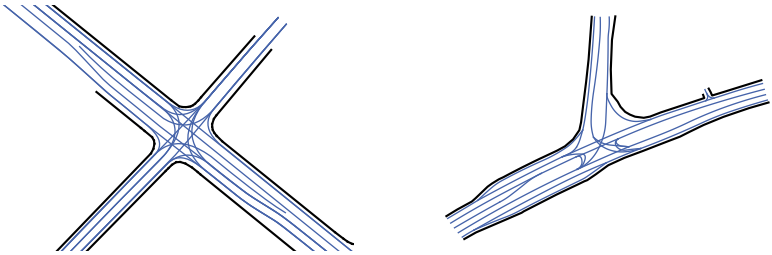


Figure 6.8 The point $P = (x_P, y_P)^\dagger$ is transformed into the Frenet frame defined by centerline $\boldsymbol{f}_{\mathcal{L}}$. As a result, the arc length s_P and the lateral distance ρ_P are obtained.

changes the position $(x, y)^\dagger$ in global Cartesian coordinates to the position $(s, \rho)^\dagger$ in local lane-dependent coordinates. The value of s gives the arc length and ρ gives the lateral distance to $\boldsymbol{f}_{\mathcal{L}}$ as illustrated in fig. 6.8. The methods provided by Wang et al. [114] are used for the realization of the transformation $\mathcal{F}_{\boldsymbol{f}_{\mathcal{L}}}$. Obviously, it is crucial to find good fits to the centerlines of the lanes in order to enable the definition of lane-dependent coordinates. For that, splines of degree two and three were applied throughout this work. The fitted curves $\boldsymbol{f}_{\mathcal{L}}$ for both examined intersections are shown in fig. 6.9. Note, that the KoPER intersection in fig. 6.9(a) is a signalized intersection with traffic lights on all sides, whereas in the inD intersection vehicles yield to vehicles on their right.

6.2.2.2 Detachment of lanes into sections

To handle splits, merges and intersections of lanes, the lanes are segmented into lane sections. The section boundaries are set to the merging and splitting positions and additionally to stopping lines. The results of the segmentation procedure when applied to the chosen KoPER and the inD intersections are displayed in figs. 6.10 and 6.11. The very same figures also describe the nomenclature used to name the lane sections in the intersections. Since the nomenclature is simple and consistent for fig. 6.10, just some example lanes have been given (the lanes starting from track 0, 1 and 2, which are not given, are named in the same way). However, the inD intersection in fig. 6.11 is more complicated and the lanes' names can not easily be derived. Therefore, all lane names are given here. In principle, the nomenclature of a lane section works similar for both intersections and uses



(a) KoPER intersection **(b)** inD intersection

Figure 6.9 Illustration of examined intersections. **(a):** Intersection in Aschaffenburg contained in the KoPER dataset. **(b):** Intersection at Neuköllner Straße, Aachen selected from the inD dataset. The blue centerlines were fitted by using splines of degree 2 and 3.

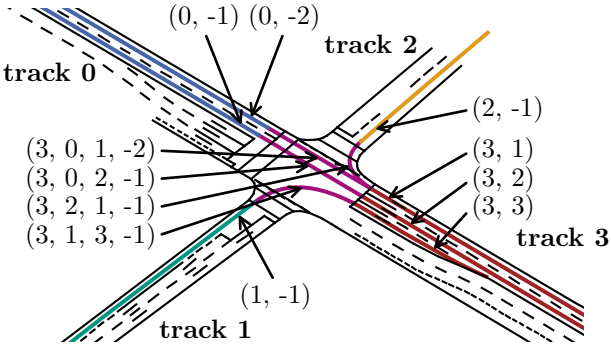


Figure 6.10 Nomenclature of lane sections in the KoPER intersection. Lane sections on the outside of the intersection are named by a 2-tuple (track, lane position) and lane sections in the intersection are named by a 4-tuple (track in, track out, lane position in, lane position out). The lanes going out of the intersection are chosen to have negative lane position values. The number for the lane position is increasing from the right to the left in the direction of travel. As an example, the image shows the lane sections starting at track 3. The lane sections of the other tracks are constructed and named equivalently. The image was firstly published by Jesenski et al. [132]; © 2019 IEEE.

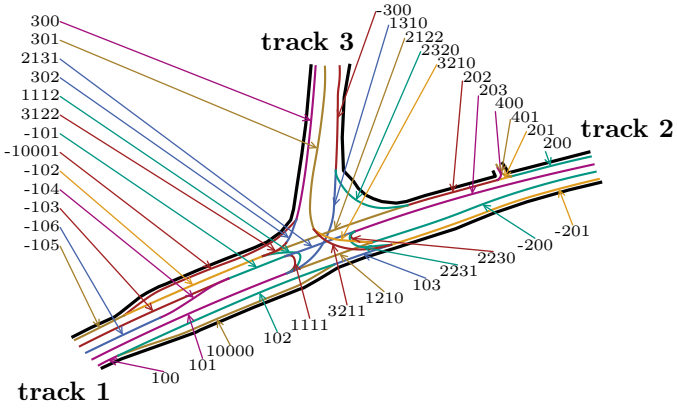


Figure 6.11 Nomenclature and construction of lane sections in the inD intersection at Neuköllner Straße, Aachen. Nomenclature is similar to the KoPER nomenclature. Lane sections on the outside of the intersection have three digits. The first digit gives the track and the next two digits give the count of the lane section within the track. Lane sections leaving the intersection are symbolized by negative numbers. Lane sections within in the intersections have four digits. The first digit gives the entering track, the second digit the track on which the section leaves the intersection and the next two digits give the counts of the connected lane sections in the entering and leaving track. Since the inD intersection is more complicated in its nomenclature, all lane sections are drawn in the image.

two properties: The driveway (track) of the lane section and its position within the driveway. Both examined intersections have four tracks which were named by integers between 0 and 3 and 1 and 4 respectively (track 4 in fig. 6.11 is not denoted but it is the small exit from track 2 on the right side of the image). More details on the nomenclature can be found in the captions of the figures.

When the lane sections will be populated (see section 6.2.2.4), relations to relevant neighboring lane sections must be considered since they stochastically influence the vehicles in the current lane section to sample. The relevant types of relations between lane sections are discussed in fig. 6.12.

In the following, it is assumed that all vehicles obey the traffic rules of the intersection. That means, only relations between adjacent lane sections, which the traffic rules allow to be populated at the same time, are considered. This has mainly an effect on the signalized KoPER intersection in fig. 6.10

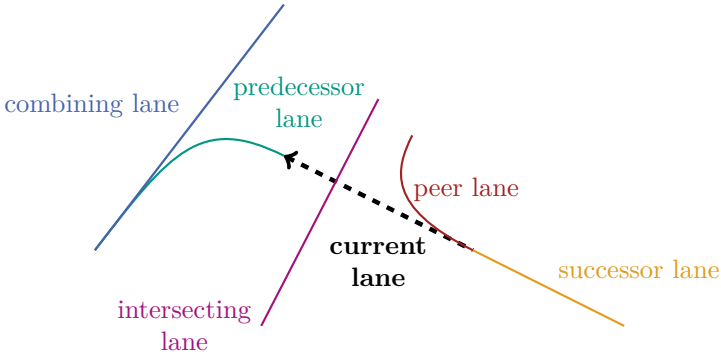


Figure 6.12 Illustration of the *current lane* section (black, dashed) and the possible relations to adjacent lane sections. The black arrow gives the direction of travel. The *predecessor lane* is the lane section before the current lane section in the direction of travel. It contains the vehicles which spatially precede the vehicles in the current lane. The *successor lane* section is accordingly placed behind the current lane. An *intersecting lane* section intersects with the current lane section at some point. Additionally, an adjacent lane section is called *peer lane* section when it possesses a common start or end point with the current lane. At last a *combining lane*, which is a peer lane section to the current lane's predecessor lane, can sometimes be relevant. The image is based on fig. 5 published by Jesenski et al. [132]; © 2019 IEEE.

since the intersection's traffic light phases determine which lane sections are allowed being entered at the same time. However, as soon as there are vehicles in the dataset not obeying these rules, this assumption must be removed. Since the aim of this section is to illustrate the concept of populating intersections and since there are no conceptual reasons, which would prevent including all relations, this simplifying assumption can be adopted with no loss of generality. More details on the traffic light phases of the KoPER intersection which determine the relevant relations of lane sections are given in appendix B.1.

6.2.2.3 Hierarchization of parameters

In order to address the challenge of global parameters which influence multiple lane sections, the partitioning of parameters into intrasectional and intersectional parameter groups shall be proposed in the following. Intrasectional parameters hereby shall have a local meaning which only influences the vehicle population within one lane section, whereas inter-

Table 6.1 Definition of the set of intersectional parameters Ψ_{global} for the KoPER and the inD intersection. $\Psi_{\text{global}}[\mathcal{L}]$ gives the parameters which are directly relevant for the usage in the local networks for respective lane section $\mathcal{L}$, see section 6.2.2.4.

	KoPER	**inD**
Ψ_{global}	$a, t_a, \mu_{(0,1)}, ..., \mu_{(3,3)}$	$\mu_{100}, ..., \mu_{-300}$
$\lvert\Psi_{\text{global}}\rvert$	35	36
$\Psi_{\text{global}}[\mathcal{L}]$	a, t_a	-

sectional parameters have a global meaning. This means they can have influence on multiple lane sections.

In the following,

- the light phase a (for a definition of KoPER's light phases see fig. B.2),
- the time t_a since the last change of the light phase,
- boolean parameters $\{\mu_{\mathcal{L}} | \mathcal{L} \in \boldsymbol{L}\}$ for the set of all lane sections $\boldsymbol{L}$ which describe if at least one vehicle can be found in lane section $\mathcal{L}$ in the current sample,

shall be the elements of the intersectional parameter group. Of course, for more detailed models a lot more parameters could be added. Since a and t_a are related to traffic lights, they are neglected for the inD intersection because no traffic lights exist at Neuköllner Straße. The sets of intersectional parameters Ψ_{global} for the inD and KoPER intersection are given in table 6.1. Because of the global meaning of these parameters, they are sampled by the use of a global BN before the intrasectional parameters for the single lanes are drawn. The drawing of the intrasectional parameters is accomplished by accessing local BNs which were trained only upon data of the respective single lanes. This procedure is sketched in fig. 6.13.

The generation of intersectional parameters is realized by drawing from the joint probability distribution

$$\Psi_{\text{global}} \sim p_{\text{global}}, \tag{6.31}$$

which is modeled by the global BN. The drawn intersectional parameters can then be utilized to infer the local intrasectional parameters Ψ_{local}. Therefore, one local BN per lane section $\mathcal{L}$ is trained to represent

$$p_{\text{local},\mathcal{L}}\left(\varphi_{\Psi_{\text{local}}} \middle| \varphi_{\Psi_{\text{global}}[\mathcal{L}]}\right), \tag{6.32}$$

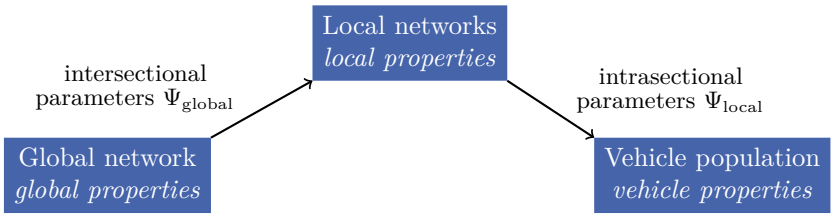


Figure 6.13 Basic structure of the Bayesian network approach. The intersectional parameters are sampled by a global network. They are then inputted in the local networks of the respective lane section and the intrasectional parameters are sampled. The intrasectional parameters directly describe the physical vehicle population in the respective lane section. Based on fig. 1 published by Jesenski et al. [132]; © 2019 IEEE.

where $\Psi_{\text{global}}[\mathcal{L}]$ gives the subset of intersectional parameters which is relevant for lane section $\mathcal{L}$. Remember again, that $\varphi_{\boldsymbol{X}}$ is a numerical assignment of the random variables in $\boldsymbol{X}$. The local, intrasectional parameters allow calculating the vehicles' properties such as position and velocity. The structure and application of the local BNs in order to populate the lanes is outlined in section 6.2.2.4. The local network for a lane $\mathcal{L}$ is only called if $\varphi_{\mu_{\mathcal{L}}} = True$. Otherwise, it is assumed that the lane is empty which then does not necessitate any call of the local network since no vehicles are to be placed.

6.2.2.4 Sampling on lane sections

In this section, the procedure to fill single lane sections by the local networks is described. The following cases must be distinguished:

- The filling of lane sections which posses neither a peer, a combining nor an intersecting lane section (case 1).
- The filling of lane sections with peer lanes (case 2).
- The both preceding cases which additionally have either an intersecting lane (cases 3 and 4), a combining lane (cases 5 and 6) or both (cases 7 and 8).

The filling of lane sections without peers (case 1) is realized similarly to the approach described in section 6.1.1. Some adaptions are however

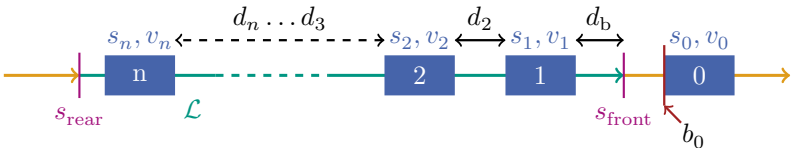


Figure 6.14 Sampling a lane section without peers (case 1). The horizontal line gives the central line of a lane. The lane does not necessarily have to be straight anymore as a change of the coordinate system into lane-dependent coordinates (s, ρ) is included. Therefore, s_i gives the arc length of the front bumper of the ith vehicle. The vertical lines named s_front and s_rear give the arc lengths of the lane section boundaries. The image was originally published by Jesenski et al. [132]; © 2019 IEEE.

indispensable because of the relations to other lane sections, the inhomogeinity of lanes and the existence of global parameters. A lane section is as before filled against the direction of travel in order to make sure that the vehicles can be accurately placed to infrastructure elements. In the examined case these are usually stopping lines in front of the respective vehicle. The extension of the approach in section 6.1.1 to the requirements of lane sections and global parameters is illustrated in fig. 6.14. As already introduced in section 6.1.1, two local BNs are used to generate the intrasectional parameters of a lane. The first of these networks samples the position of the first car in the lane section. The second network samples the next vehicles conditioned on the respective predecessor vehicle. The joint probability distribution modeled by the first network is adapted to

$$v_1, d_\mathrm{b} \sim p_1 \left(\varphi_{v_1}, \varphi_{d_\mathrm{b}} \middle| \varphi_{b_0}, \varphi_{v_0}, \varphi_{\Psi_\mathrm{global}[\mathcal{L}]} \right). \tag{6.33}$$

Since a lane section has a predecessor lane section, it is in contrast to (6.1) necessary to condition on properties b_0, v_0 of the last vehicle in this predecessor lane section. It is of course a valid assumption that no predecessor vehicle exists in the predecessor lane section. For this case $\varphi_{b_0} = \mathrm{NaN}$ and $\varphi_{v_0} = \mathrm{NaN}$ are encoded as extra bins in the network. Additionally, the conditioning on the global parameters $\Psi_\mathrm{global}[\mathcal{L}]$, which are relevant for lane section $\mathcal{L}$, is added. Note that the lane section is also strongly dependent on $\mu_\mathcal{L} = True$ since otherwise the networks are not even called. If the back bumper b_0 of vehicle 0 is in $\mathcal{L}$ (that is $b_0 < s_\mathrm{front}$), the starting point for the distance d_b is set to $s_\mathrm{front} = b_0$. This guarantees that there is no overlap between vehicle 0 and vehicle 1.

The second distribution (6.2) is modified to sample the $i+1$th vehicle by a distribution

$$p_{\text{next}}\left(\varphi_{v_{i+1}}, \varphi_{d_{i+1}} \middle| \varphi_{v_i}, \varphi_{v_{i-1}}, \varphi_{d_i}, \varphi_{s_i}, \varphi_{\Psi_{\text{global}}[\mathcal{L}]}\right). \tag{6.34}$$

Due to the inhomogeneity of the lanes, it becomes necessary to condition on the arc length s_i of the vehicles. Of course, also the relevant intersectional parameters are included in the condition part of the distribution.

Note that for the sake of simplicity and in order to keep the size of the BNs small (remember that inference might scale exponentially with the network size), the length l_i and width w_i of the sampled vehicles are neglected in the probability distributions. Consequently, constant values will be used in the simulations. Analogously, the parameters in (6.2) which are related to vehicles in the lane to the right are also neglected. In total, the front bumper position s_{i+1} of the $i+1$th vehicle is then generated (analogously to (6.3)) by

$$\varphi_{s_{i+1}} = \begin{cases} \varphi_{s_i} - \varphi_{d_{i+1}} - \varphi_{l_i} & \text{for } i \geq 1 \\ \min\left(s_{\text{front}}, \varphi_{b_0}\right) - \varphi_{d_{\text{b}}} & \text{for } i = 0 \end{cases}. \tag{6.35}$$

The sampling of new vehicles is stopped for $\varphi_{s_{i+1}} < s_{\text{rear}}$ or if the second networks returns NaN for d_{i+1} or v_{i+1}. These missing values, encoded by an extra bin state, express that no further successor exists (this is not allowed for the first network due to the global parameter $\mu_{\mathcal{L}} = True$ ensuring that at least one vehicle must exist).

For the sampling of lane sections with related peer lanes (case 2), a modified approach becomes necessary since the correlations to vehicles in peer lanes have to be taken into account. For instance, vehicles in lane section $(3, 0, 1, -2)$ of the KoPER intersection can block space on lane section $(3, 2, 1, -1)$ especially directly after the lane split when the distance between the lane sections' center lines is still small. Hence, the concept of *virtual vehicles*, which allows considering this blocked space, shall be introduced in the following. Afterwards, the application of these virtual vehicles to the sampling of vehicles will be discussed.

The creation of virtual vehicles is sketched in fig. 6.15. Virtual vehicles of a lane $\mathcal{L}$ are created by projecting the vehicles in $\mathcal{L}$'s peer lanes onto $\mathcal{L}$'s

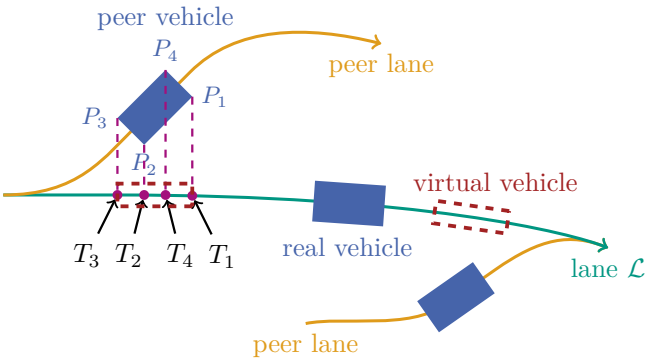


Figure 6.15 Creation of virtual vehicles: Projection of real vehicles in peer lanes on their virtual vehicle representations on the current lane $\mathcal{L}$. The virtual vehicles block space on $\mathcal{L}$. Image based on fig. 7 published by Jesenski et al. [132]; © 2019 IEEE.

central line. The points $\{P_i | i = 1, ..., 4\}$ describe the position of the peer vehicles' bounding boxes in global coordinates. The projection of the peer lane vehicle can thus be accomplished based on (6.30) and by executing

$$T_i = Proj_{\mathcal{L}} \left[\mathcal{F}_{\boldsymbol{f}_{\mathcal{L}}} (P_i) \right] = Proj_{\mathcal{L}} \left[\begin{pmatrix} s_{P_i} \\ \rho_{P_i} \end{pmatrix} \right] = \begin{pmatrix} s_{P_i} \\ 0 \end{pmatrix} \forall i \in \{1, ..., 4\} \,. \quad (6.36)$$

Here, $Proj_{\mathcal{L}} [\bullet]$ is the projection of a point to the central line of lane section $\mathcal{L}$. The points T_i thus are the projected points which define the virtual vehicle. The arc lengths of the front and rear bumper of the virtual vehicle are specified by the extremal values of the arc lengths of the points T_i. Since the blocking of an arc length interval by a virtual vehicle only makes sense if the projected vehicle is near to the current lane, an additional condition of

$$\min_{i \in \{1,2,3,4\}} (\|T_i - P_i\|_2) \leq L \quad (6.37)$$

must be fulfilled. The threshold L gives the minimal distance to $\mathcal{L}$ a vehicle on a peer lane section must undercut to create a virtual vehicle. For the creation of virtual vehicles, the order of sampling the lane sections is substantial. The sampling order used within this work will be discussed in section 6.2.2.5.

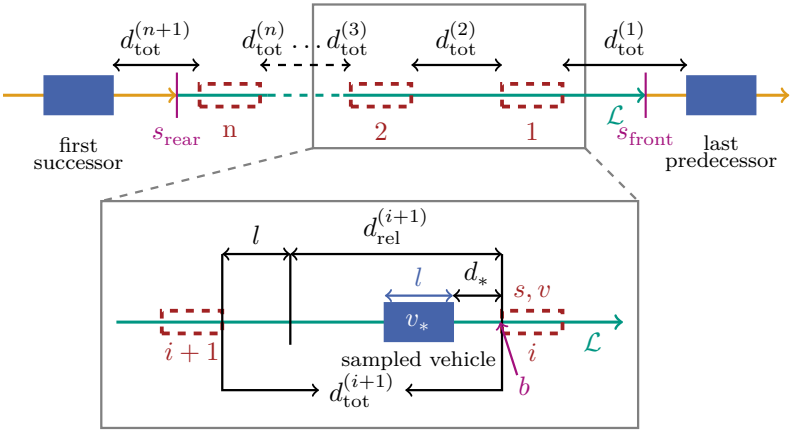


Figure 6.16 Placement of real vehicles in gaps between virtual vehicles and the last and first vehicles of the preceding and successing lane sections (case 2). The image was firstly published by Jesenski et al. [132]; © 2019 IEEE.

The sampling model is applied after the generation of all virtual vehicles in a lane section $\mathcal{L}$. Since the virtual vehicles block arc length intervals, the lane must be filled by placing real vehicles in the gaps between the virtual vehicles. This procedure is illustrated in fig. 6.16. The gaps in a lane section are filled against the direction of travel. The algorithm starts with the gap between the last predecessor and the 1st virtual vehicle. The procedure for the gap between the ith and $i+1$th virtual vehicle acts by the following scheme:

1. The preceding vehicle of the gap is determined. This is usually a virtual vehicle or real vehicle in the case of last predecessor (here: the ith virtual vehicle). From now on, this vehicle is denoted as *current vehicle*.

2. The arc length b at the rear bumper and the velocity v of the current vehicle are taken.

3. The distance $d_{\text{tot}}^{(i+1)}$ between the current vehicle and the next successor which ends the gap is calculated. This is usually a virtual vehicle or a real vehicle in the case of first successor, see fig. 6.16 (here: $i+1$th virtual vehicle). The current vehicle's distance d_{pre} to the real or

virtual vehicle, which precedes it in the direction of travel, is taken. Additionally, the velocity v_{pre} of this preceding vehicle is examined.

4. These values can be used to draw samples from a BN representing the joint distribution

$$p\left(\varphi_{v_*}, \varphi_\zeta \middle| \varphi_b, \varphi_v, \varphi_{d_{\text{pre}}}, \varphi_{v_{\text{pre}}}, \varphi_{d_{\text{tot}}^{(i+1)}}, \varphi_{\Psi_{\text{global}}[\mathcal{L}]}\right). \tag{6.38}$$

The distance to the next vehicle which now shall be sampled is then given by

$$d_* = \zeta \cdot d_{\text{rel}}^{(i+1)}. \tag{6.39}$$

The parameter $d_{\text{rel}}^{(i+1)} = d_{\text{tot}}^{(i+1)} - l$ is the distance to the next (virtual) vehicle minus the length l of the vehicle to sample.

5. Check the conditions

$$\varphi_{d_{\text{tot}}^{(i+1)}} < l, \tag{6.40}$$

$$\varphi_\zeta = \text{NaN}, \tag{6.41}$$

$$\varphi_b - \varphi_{d_*} < s_{\text{rear}}. \tag{6.42}$$

The first condition checks if there is enough space left in the current gap for the next sampled vehicle to be placed in front of the gap-ending (virtual) vehicle without any overlap. The second condition considers the encoding NaN which signalizes that no further vehicle shall be placed in the current gap and the third condition checks if the sampled vehicle is still located in the current lane. If none of these conditions holds then set the next real vehicle with velocity v_* at $s_* = b - d_*$. Define this new vehicle to be the new *current vehicle* and reiterate the sampling process beginning with step 2. However, if any of the conditions holds, then reject the drawn sample and check if another gap succeeds the current gap within $\mathcal{L}$ (here: between virtual vehicle $i+1$ and virtual vehicle $i+2$ (or first successor respectively)). If true, then select this next succeeding gap and start filling it by reiterating the whole process starting at 1. If no further gap exists, the sampling is aborted for $\mathcal{L}$ in total.

Remember that the sampling in $\mathcal{L}$ is only started if $\mu_\mathcal{L} = True$. However, if the sampling started at least one vehicle is demanded. Therefore, if

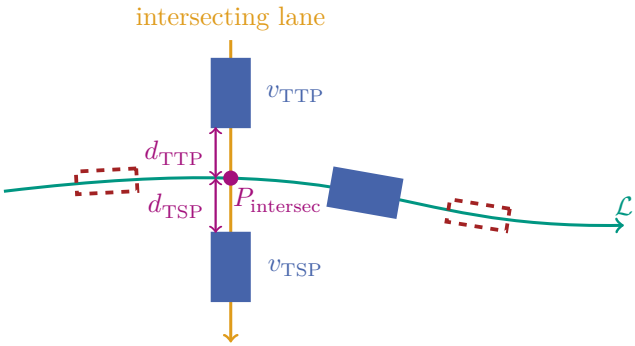


Figure 6.17 Illustration of an intersection of lane sections. The nearest vehicles in the intersecting lane are considered and the ***T**ime **T**o intersection **P**oint* (TTP) as well as the ***T**ime **S**ince intersection **P**oint* (TSP) is calculated by using (6.43) and (6.44). The figure was firstly published by Jesenski et al. [132]; © 2019 IEEE.

the whole procedure does not produce any sampled vehicle since for each iteration any of the conditions in (6.40) to (6.42) holds, then repeat the sampling procedure for the whole lane section until at least one vehicle is sampled in $\mathcal{L}$ or a maximal number of tries has been reached. Note that if no *last predecessor* vehicle exists, its values are all set to NaN and $s_* = s_{\mathrm{front}} - d_*$.

For lane sections with intersecting lanes (cases 3, 4, 7, 8), it is important that the last intersecting vehicle after the intersection point P_{intersec} and the first intersecting vehicle before P_{intersec} are considered when populating a lane section $\mathcal{L}$. Figure 6.17 shows the situation. By using the parameters of these vehicles as described in fig. 6.17, it is possible to calculate the *time since intersection point* (TSP) and the *time to intersection point* (TTP) by

$$TTP = \frac{d_{\mathrm{TTP}}}{v_{\mathrm{TTP}}}, \tag{6.43}$$

$$TSP = \frac{d_{\mathrm{TSP}}}{v_{\mathrm{TSP}}}. \tag{6.44}$$

When a lane section with an intersecting lane is filled, the relevant intersecting vehicles are considered by additionally conditioning the probability distributions given in (6.33), (6.34) and (6.38) on TTP and TSP.

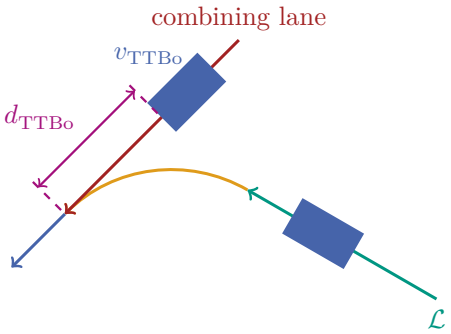


Figure 6.18 Illustration of a combining lane scenario. The first vehicle in the combining lane is used to calculate TTBo by applying (6.45).

Lane sections with combining lanes (cases 5, 6, 7 and 8) require, analogously to intersecting lanes, to consider the first vehicle on a combining lane section. This is especially important when vehicles in lane section $\mathcal{L}$ must yield to vehicles in the combining lane section since then the appearance of a vehicle in this lane section has great influence on the behavior of a vehicle in $\mathcal{L}$. For instance, this case arises for lane sections 302 and 3122 in relation to the combining lane section 2122 at Neuköllner Straße since vehicles in 302 and 3122 must yield to vehicles in 2122 and therefore strongly reduce their velocity if necessary. Figure 6.18 gives the situation and describes d_{TTBo} and v_{TTBo}. By putting these parameters into

$$TTBo = \frac{d_{\mathrm{TTBo}}}{v_{\mathrm{TTBo}}}, \tag{6.45}$$

the *time to boundary* (TTBo) is determined. TTBo is connected to the scene generation procedure by simply putting it into the conditioning part of (6.33), (6.34) and (6.38).

6.2.2.5 Sampling order

In the process of filling the lane sections, vehicles in related lane sections are considered as described above. Therefore, the order of filling the lane sections is very important. Since a lane section always is conditioned on its predecessor lane section, the filling procedure starts by sampling the lane sections at the front of the examined road topology since they do not

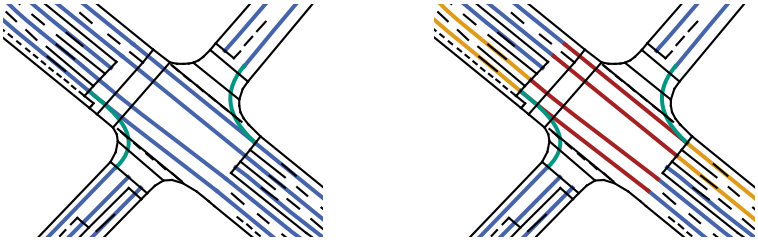


Figure 6.19 Illustrations for the lane sections which are allowed to be traversed in light phase $a = 1$ of the KoPER intersection (compare fig. B.2), when the traffic lights at track 0 and track 3 are green. **(a)** distinguishes lane sections which are not conditioned on related peer sections (blue) and lane sections whose filling procedure is dependent on peer sections (green). **(b)** shows the order of filling the lane sections. First of all, the blue lanes are filled, followed by the red ones and the yellow ones. The green peer related lane sections are filled at the end. The images were published by Jesenski et al. [132]; © 2019 IEEE.

have predecessor sections. Afterwards, the successor lane sections are filled iteratively until the rear lane sections are reached. During this process, the lanes which possess related peer lane sections are skipped at first. When a lane section $\mathcal{L}$ depends on related intersecting or combining lane sections, then these must be populated before $\mathcal{L}$. The respective parameters are added into $\mathcal{L}$'s distribution as already described. At the end, the lane sections which must be conditioned on related peer lane sections are filled. An example for the KoPER intersection is given in fig. 6.19.

6.2.3 Dataset & discretization

For the training of the BNs which represent the introduced distributions, the KoPER dataset as well as Neuköllner Straße in the inD dataset were used. More details on the datasets are given in appendix B. In order to train the BNs on the datasets, the continuous dataset parameters must firstly be discretized. For this discretization the ranges of the dataset parameters are divided into equidistantly spaced bins. The minimal and maximal values of these ranges are used to determine the middle value of the respective extremal bins. The parameters used in the BNs and the number of bins into which they are discretized can be found in table 6.2. Since the KoPER

Table 6.2 The parameters used to train the Bayesian networks, their meaning and the number of bins used to discretize them. The light phase a is a discrete value and does not have to be discretized. The table is based on tab. 1 published by Jesenski et al. [132], © 2019 IEEE.

Parameter	Description	Bin Counts
	intersectional parameters	
a	light phase	discrete
t_a	time since last change of light phase	20
	parameters for lane sections without peers – first vehicle	
v_1	velocity of first vehicle	20
d_b	distance of first vehicle to boundary s_front	40
b_0	rear bumper of last vehicle in predecessor lane	40
v_0	velocity of last vehicle in predecessor lane	20
	parameters for lane sections without peers – following vehicles	
v_{i+1}	velocity of successor	20
v_i	velocity of current vehicle	20
s_i	arc length of current vehicle	40
v_{i-1}	velocity of predecessor of current vehicle	20
d_{i+1}	distance to successor of current vehicle	40
d_i	distance to predecessor	40
	parameters for lane sections with peers	
v_*	velocity of successor to sample	20
ζ	relative distance to real successor vehicle	40
b	arc length of rear bumper of current vehicle	40
v	velocity of current vehicle	20
d_pre	distance to predecessor of current vehicle	40
v_pre	velocity of predecessor of current vehicle	20
$d_\mathrm{tot}^{(i+1)}$	dist. to virt. successor of current vehicle	40
	parameters for lane sections with intersecting lane sections	
TTP	time to intersection point	10
TSP	time since intersection point	10
	parameters for lane sections with combining lane sections	
TTBo	time to boundary	10

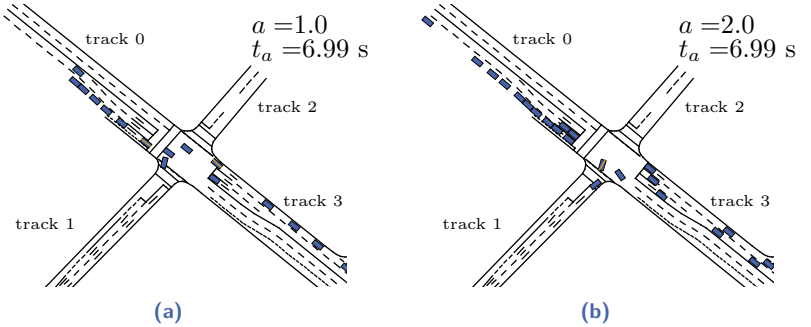


Figure 6.20 Scenes in the KoPER intersection sampled using BNs. The orange curves in both images represent virtual vehicles. Most vehicles in the dataset are driving in the direction between track 0 and track 3. Accordingly, way more vehicles are placed on these tracks than on track 1 and 2. **(a)** shows a scene about 7 s after the traffic lights entered traffic light phase 1. That means, the vehicles in track 0 and track 3 are allowed to drive into the intersection. The generated scene is consistent with that. Since the traffic light change took place only 7 s before, a traffic jam on track 0 is in a dissolving state at the moment. The reason is that the light phase $a = 2$ takes place before light phase $a = 1$, here vehicles from track 0 and track 3 are not allowed to enter the intersection. **(b)** shows such a scene with $a = 2$. It can be seen that the vehicles from track 0 and track 3 are waiting at their red traffic lights whereas vehicles from track 1 are entering the intersection. By chance, no vehicles are existent on track 2.

dataset only provides 6 min 28 s of data, all data are used for training. Since 1.83 h of data are available for Neuköllner Straße in the inD dataset, the relevant inD data are split into 90 % of training data and 10 % of test data.

6.2.4 Evaluation

The model described in section 6.2.2 is used to sample scenes in both examined intersections. Exemplary results are given in figs. 6.20 and 6.21. Visually, the generated vehicle positions look realistic. The velocities of the vehicles in the example scenes for the KoPER intersection (velocities not illustrated in fig. 6.20) show that the cars which are allowed to drive into the intersection due to the traffic light phase are moving fast whereas the rest of the cars in front of red traffic lights are moving slowly or are stopped. Additionally, the images illustrate the dissolution (fig. 6.20(a)) and formation (fig. 6.20(b)) of traffic jams in track 0. This indicates, that the conditioning on global parameters is working. In fig. 6.21, the vehicle in lane section 302 stops and yields to the vehicle in lane section 2122 which

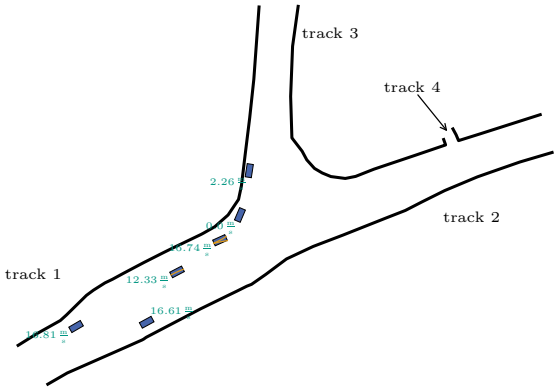
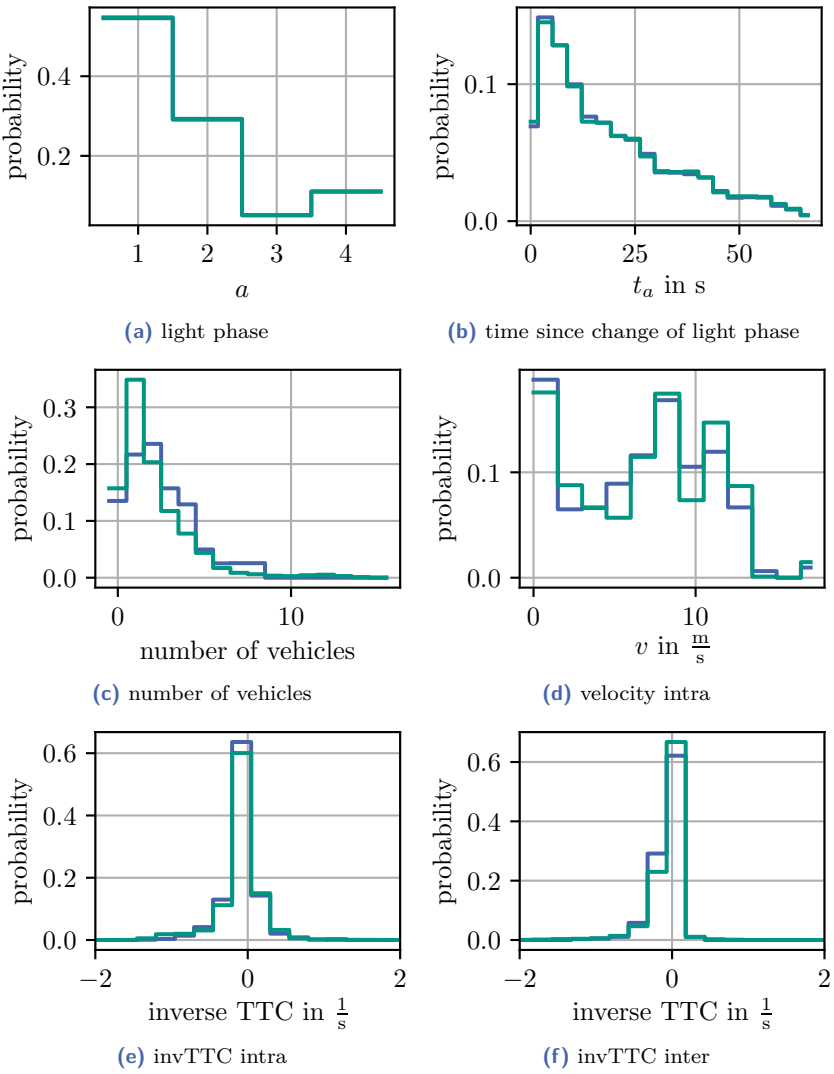


Figure 6.21 Scene sampled for Neuköllner Straße. This intersection is not signalized. Vehicles from the track 3 must yield to vehicles driving between track 1 and track 2. The vehicle in lane 302 accordingly yields to the vehicle which has driven from track 2 to track 1. It therefore is in standstill with a velocity of 0.0 $\frac{\text{m}}{\text{s}}$. The orange curves again give the lane section intervals blocked by virtual vehicles. Most parts of track 2 were not in the field of view of the camera which was used to collect the dataset. Therefore, no vehicles can be sampled in this region.

is caused by the conditioning on TTBo.

To quantitatively assess the sampling procedure, several metrics were evaluated on both datasets and on 20 000 generated samples respectively. First of all, the signalized KoPER intersection requires an assessment of the intersectional parameters a and t_a. The results are given in figs. 6.22(a) and 6.22(b). Obviously, there is a remarkable accordance between dataset and sampled scenes. The used metrics for intrasectional properties are the number of vehicles per lane section (*number of vehicles*), the inverse TTC (see section 4.3) between vehicles in the same lane section (*invTTC intra*), the velocity distribution of all vehicles in a lane section (*velocity intra*) and the inverse TTC between vehicles at a transition $(\mathcal{L}_1, \mathcal{L}_2)$ between two lane sections (*invTTC inter*). The successor vehicle of the TTC calculation resides in lane section $\mathcal{L}_1$ and the predecessor is located in lane section $\mathcal{L}_2$. The number of vehicles and both invTTCs are metrics which were not directly learned by the BNs. The velocity distribution however is included in the training parameters. The metrics are evaluated separately for all relevant lane sections, respectively all transitions between lane sections



(a) light phase **(b)** time since change of light phase

(c) number of vehicles **(d)** velocity intra

(e) invTTC intra **(f)** invTTC inter

Figure 6.22 Comparison of distributions of several metrics for the KoPER dataset. The blue curves show the distribution in the dataset and the green curves show the distributions in the 20 000 generated samples. **(a)**, **(b)** give the distributions of the intersectional parameters. The metrics in **(c)**, **(d)**, **(e)** show the results for lane section (3, 1) and $a = 1$. **(f)** shows the result for the transition between (3, 1) and (3, 0, 1, -2) for $a = 1$. The image is based on fig. 13 published by Jesenski et al. [132]; © 2019 IEEE.

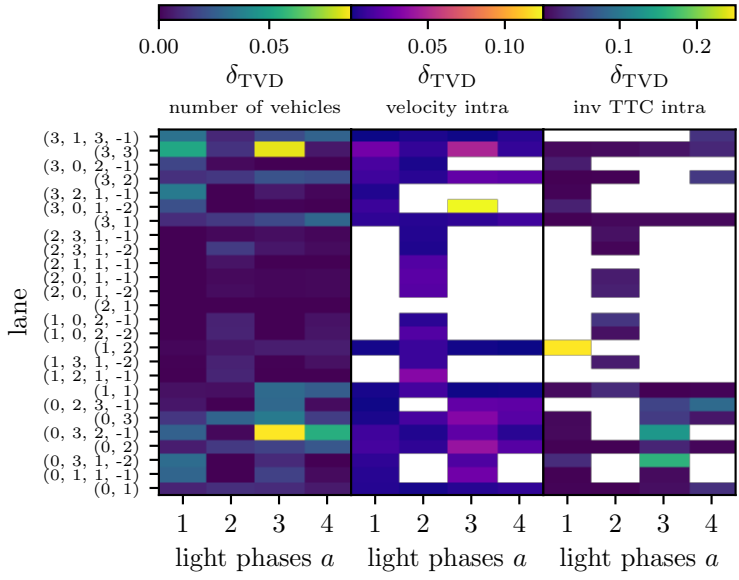
for invTTC inter. For KoPER, the results are additionally splitted for all values of the light phase a. Some exemplary results for the KoPER dataset are illustrated in figs. 6.22(c) to 6.22(f). These exemplary results for the intrasectional metrics also suggest that the generated samples are in accordance with the dataset.

For all assessed lane sections and light phases, the "distance" between the empirical probability distributions of the dataset and the generated samples is quantified by using the *total variation distance* (TVD), normalized to the number of bins. This procedure is defined by

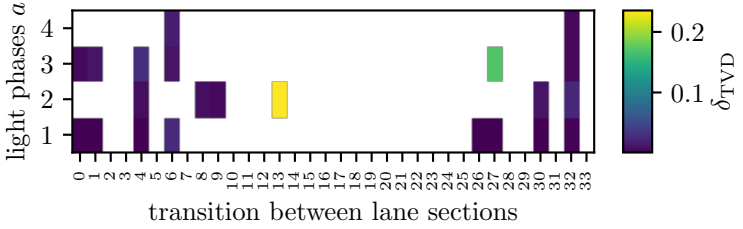
$$\delta_{\mathrm{TVD}} = \frac{1}{2b} \sum_{i=1}^{b} |p_i - q_i| \,. \tag{6.46}$$

p and q are the distributions in the dataset and the samples (as exemplary shown in fig. 6.22). b is the bin count of the distributions, corrected by the number of bins where both q_i and p_i are zero. The results for the KoPER dataset are given in fig. 6.23 and the inD results are shown in fig. 6.24.

In the KoPER dataset, the light phases endure for different durations. In total, $a = 1$ lasts for 200 s, $a = 2$ for 100 s, $a = 3$ for 20 s and $a = 4$ for 40 s. Hence for the KoPER dataset, the worst values for the TVD are expected to be in light phases 3 and 4 because the least training data are available here. This is verified by fig. 6.23(a), which gives an overview about the TVD values calculated for all mentioned intrasectional metrics on all (lane, a) combinations. Figure 6.23(b) gives the same information for the invTTC inter metric. It can be seen, that at transition number 13 which corresponds to the transition $((1,2),(1,0,2,-2))$ an outlier occurs. In this context it is necessary to note that only few vehicles are driving in the involved lane sections and therefore only few training data are available at this transition (this low number of vehicles in lane section $(1,2)$ also causes the maximum value for TTC intra in fig. 6.23(a)). Figure 6.23(a) shows, that the number of vehicles coincides very well at light phases where the respective lanes are not allowed to contain vehicles as a result of the traffic rules (i.e. most of the (lane, a) combinations which correspond to white spaces in the TVD of the velocity intra metric correspond to very low values for the TVD of the number of vehicles metric). The reason is, that for a high probability to have no vehicles located in a lane section, the statistics are determined strongly by the intersectional parameters $\mu_{\mathcal{L}}$, which means that for this cases the metric is learned directly by the global network and not generated indirectly by the intrasectional networks. Additionally, there



(a) intrasectional metrics



(b) invTTC inter

Figure 6.23 Results for the KoPER dataset. **(a)** shows the heatmaps which illustrate the results for the TVD for the intrasectional metrics separately for all combinations of relevant lane sections and phases. Lower values are better.White spaces mean that no data were available for the (lane, a) combination. The most white spaces appear for invTTC intra since the calculation of a TTC needs two vehicles. The lane sections directing out of the intersections were neither sampled nor evaluated since the field of view of the sensors used to obtain the dataset did not cover them sufficiently. **(b)** shows the results for the TVD for invTTC inter. Lower values are better. In total, 34 transitions are allowed between the 26 relevant lanes. Note that for the sake of clarity, the transitions just were enumerated by integers since denoting the respective 2-tuples of the lane sections in a transition would require a lot of space. The large outlier at lane transition number 13 is caused by too few training data. White spaces mean no data were available. Both images were firstly published by Jesenski et al. [132]; © 2019 IEEE.

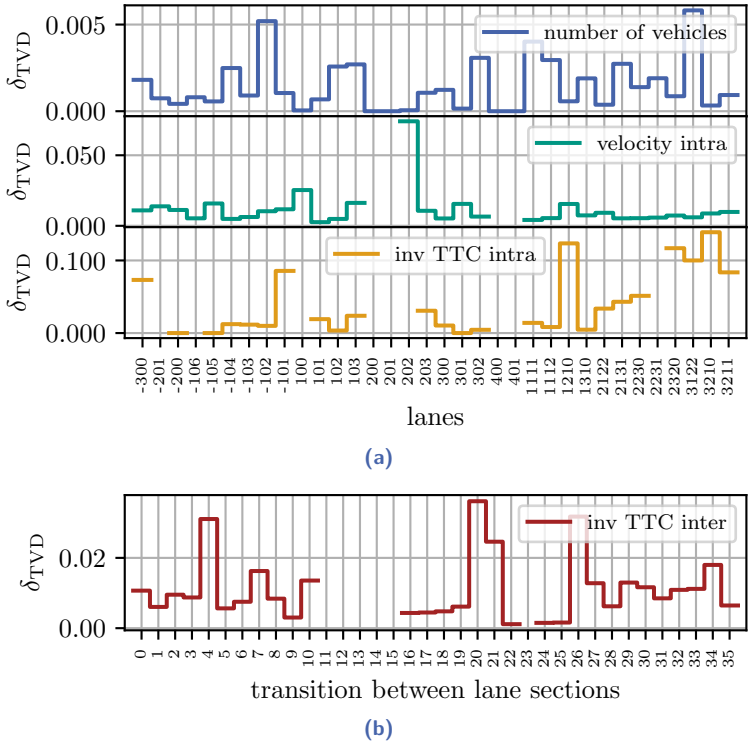


Figure 6.24 Results for Neuköllner Straße (inD). **(a)** shows the TVD results for the intrasectional metrics. **(b)** gives TVD results for invTTC inter. The transitions between lane sections are again just enumerated by numbers. The discontinuations in the histograms correspond to missing data.

are some (lane, a) combinations for which the traffic rules do not allow the filling of the lane section, but nevertheless the TVD of the velocity intra metric contains some values for the combination. An example is (3, 1, 3, -1) for $a \in \{1, 2, 3\}$. This is caused by humans driving slightly over the stopping lines of a red traffic light and waiting in the intersection. However such vehicles stand still and wait until the traffic lights turn green. Thus, the velocity distributions for these cases are limited to a short range in the surrounding of $v \approx 0 \frac{\text{m}}{\text{s}}$ and consequently the values for the TVD of the velocity metric often are quite low here.

Since Neuköllner Straße is not signalized, the TVD results for the in-

trasectional metrics in fig. 6.24(a) and for the intersectional metric in fig. 6.24(b) are only distinguished by lane section and not by light phase. The TVD values for the number of vehicles and velocity intra are generally lower than the respective values for the KoPER dataset. This was expected, since the inD dataset contains more data than the KoPER dataset which could be used to train the networks[2]. Especially TVD values for the number of vehicles metrics are very low. The values for velocity intra are slightly higher, but again the outliers can be explained. For example the outlier at lane section 202 can be explained by few data since vehicles on track 2 were not thoroughly recorded by the drone since most of the track is outside of the viewing angle of the camera. Therefore, incoming vehicles are detected late. For the same reasons no values for tracks 200 and 201 are given at all. The same explanation also holds for lane section 100. The values for invTTC intra are comparable to the respective values of the KoPER dataset. The reason for this lies in the low traffic density in the inD data. Hence, within most lane sections at most one vehicle is located at the same time point. Therefore, only comparatively few indirect training data (as mentioned TTC is not learned directly) for TTC were available. The fits of invTTC inter are better and the respective δ_{TVD} are rather low.

6.3 Comparison of SPNs and BNs

After having extended the incremental roadway population models from the original approach as proposed by Wheeler et al. [119] to the sampling of more complex road topologies, this section discusses how to use SPNs to improve the efficiency of the models. For doing so, the BN and SPN training methods described in section 6.1.2 and section 6.1.3 are applied to learn the distributions in (6.1) and (6.2) on the basis of the highD dataset (appendix B.2). The distributions are then used to generate highway scenes. Section 6.3.1 describes the preprocessing of the highD dataset, section 6.3.2 discusses the hyper-parameters of the SPN learning algorithms and section 6.3.3 evaluates the generated highway scenes. For that, the SPN-learning algorithms' hyper-parameters are optimized in section 6.3.3.1 and the different SPN algorithms are compared to each other and to the usage of a BN in section 6.3.3.2. This section was already published by Jesenski et al. [135]; © 2020 IEEE.

[2] inD: 1.83 h at Neuköllner Straße; KoPER: 6 min28 s, compare appendix B.

Table 6.3 Bin numbers for the discretization of vehicle velocity v, vehicle width w, vehicle length l, longitudinal distance d between vehicles, distance to boundary d_b, $d_\mathrm{right}^{(\mathrm{ego})}$ and d_right. For definitions of $d_\mathrm{right}^{(\mathrm{ego})}$ and d_right see fig. 6.1. Table taken from Jesenski et al. [135]; © 2020 IEEE.

feature	v	w	l	d	$d_\mathrm{right}^{(\mathrm{ego})}$	d_right	d_b
number of bins	30	10	10	80	10	10	80

6.3.1 Data processing

As already mentioned before, the highD dataset (appendix B.2) shall be used to train the model described in section 6.1.1. However, before the training of the probability distributions can be started, the dataset must be processed as described in the following. First of all, the positions of the vehicles in the highD dataset are transformed into lane-related Frenet coordinates. Afterwards, features comprised in the probability distributions (6.1) and (6.2) which are missing in the dataset are calculated. For the sake of sampling speed, discrete SPNs are trained. Therefore, the calculated features are discretized into equidistantly distributed bins and the calculated and discretized feature tuples are saved. They can now be used for the training procedure. Due to the feature ranges and diverse importance of the features to the sampled scenes, different bin numbers are used for the different features. The bin numbers are given in table 6.3.

6.3.2 Hyper-parameters for SPN learning

An overview about the hyper-parameters which are required by the SPN learning algorithms is presented in table 6.4. Remember, that details about the learning algorithms were already given in section 6.1.3.

It is necessary to specify the minimal number of instances κ_min which is required that an operation/instance split is allowed. The assessment of the independency of variable subsets relies on dependency measures with a usual range of 0 to 1. Therefore, one must decide for which threshold of significance $\alpha \in [0, 1]$ the algorithms shall consider a variable subset to be independent. For MSPN, the Laplace smoothing, applied on the data when fitting the leaves, must be defined by choosing the smoothing parameter Δ. SSPN and OSSPN work by augmenting an already existing SPN. Hence, a seed SPN $\mathcal{N}_0$ has to be selected. The amount of recent samples ι onto

Table 6.4 The hyper-parameters and settings needed for the training algorithms. The table was originally published by Jesenski et al. [135]; © 2020 IEEE.

hyper-parameter	description	necessary for
$\kappa_{\min}$	min. number of instances for split	LSPN, MSPN, SSPN, OSSPN
α	thresh. of significance	LSPN, MSPN, SSPN, OSSPN
Δ	smoothing parameter	MSPN
$\mathcal{N}_0$	seed SPN	SSPN, OSSPN
ι	most recent samples used for training	OSSPN
Λ	values per batch	OSSPN
σ	type of leaf distribution	LSPN, SSPN, OSSPN

which OSSPN trains when a new batch of data arrives is another important hyper-parameter. Besides, the size Λ of the batches given to OSSPN must be defined. At last, the type σ of leaf distribution p_u used when applying LSPN, SSPN and OSSPN must be chosen.

6.3.3 Evaluation

In a first step, the hyper-parameters for the SPN training algorithms must be chosen (see section 6.3.3.1). Afterwards, the chosen hyper-parameter combinations are used to compare the learning algorithms against each other and against a BN-baseline (section 6.3.3.2). For all experiments 90 % of the data were used for the training of the networks, whereas 10 % of the same were used as test dataset in order to calculate the average log-likelihoods $\mathfrak{Li}_{\log}$.

6.3.3.1 Intra-algorithmic hyper-parameter selection

All learning approaches use RDC [68] to calculate the independency of variable subsets when executing a variable split. For LSPN, SSPN and OSSPN, the k-means algorithm is used to cluster the instances. For MSPN a version using k-means, which is denoted as MSPN_{km} is implemented. Additionally, the original MSPN version called MSPN_{rdc}, for which k-means is applied on the feature space which was transformed by using the

non-linear transformations of the RDC-pipeline, is considered. Categorical distributions are used for the leaf nodes of LSPN, SSPN and OSSPN. The standard values $\Delta = 1$ and $\Lambda = 100\,000$ are used for Laplace smoothing of MSPN-leaves and the batch size of OSSPN. For the seed SPNs $\mathcal{N}_0$ of SSPN and OSSPN a fully factorized product node as proposed in [25] is applied. Note that since categorical leaves and RDC are used for this thesis' version of LSPN, the main difference to MSPN_{km} is the application of Laplace smoothing. In the following paragraphs, different values for the remaining hyper-parameters α, $\kappa_{\min}$ and ι are evaluated.

For the hyper-parameter α, the values $\alpha \in \{0.2, 0.3, 0.4\}$ shall be assessed in this paragraph for all training methods. During this assessment, the

Table 6.5 Results of the comparison of different values of α for all learning algorithms. $\mathfrak{Li}_{\text{log}}$ is the log-likelihood averaged over all data samples with a probability $\mathcal{N}(\boldsymbol{x}) > 0$ and over the networks for the following vehicles trained for the different lanes. q gives the share of data samples with $\mathcal{N}(\boldsymbol{x}) = 0$, E gives the number of edges averaged over the following-vehicle-SPNs for the different lanes and t_{train} shows the training times. $\alpha = 0.4$ is selected for OSSPN. For the rest, $\alpha = 0.3$ is chosen. The bold printed numbers give the best values of its class and training method. The table has already been published by Jesenski et al. [135]; © 2020 IEEE.

algorithm	α	$\mathfrak{Li}_{\text{log}}$	q	E	t_{train} in s
LSPN	0.2	**−14.81**	2.72×10^{-6}	1873.3	11 808
	0.3	−14.93	1.17×10^{-6}	907.0	7957
	0.4	−15.20	$\mathbf{3.88 \times 10^{-7}}$	**533.7**	**6031**
MSPN_{km}	0.2	**−14.81**	0	1873.3	11 805
	0.3	−14.93	0	907.0	7950
	0.4	−15.20	0	**533.7**	**6022**
MSPN_{RDC}	0.2	**−15.14**	0	1632.0	16 035
	0.3	−15.14	0	1314.8	14 206
	0.4	−15.22	0	**1150.2**	**13 199**
SSPN	0.2	**−14.81**	2.33×10^{-6}	2554.8	13 914
	0.3	−14.97	1.17×10^{-6}	1277.0	10 060
	0.4	−15.23	$\mathbf{3.88 \times 10^{-7}}$	**714.7**	**7903**
OSSPN	0.2	**−14.83**	1.61×10^{-3}	2557.5	15 138
	0.3	−14.96	6.75×10^{-4}	1341.5	11 184
	0.4	−15.23	$\mathbf{3.91 \times 10^{-4}}$	**752.7**	**7071**

constant values $\kappa_{\min} = 0.5\,\%$ and $\iota = 100\,000$ are set. The results of the analysis are displayed by table 6.5. For LSPN, MSPN_{km}, MSPN_{rdc} and SSPN $\alpha = 0.3$ is selected, since this seems to be the best trade-off between accuracy (high values for $\mathfrak{Li}_{\log}$) and the inference time which is linearly dependent on the number of edges E. For OSSPN the values for q are rather high. Hence, OSSPN considers a lot of data samples to be impossible and to have a probability value of zero when the likelihood is calculated. The reason is, that only the most recent 100 000 data samples were considered for the training of the parameters of OSSPN. In contrast, q disappears for both versions of MSPN. The reason should be the usage of Laplace smoothing in the MSPN leaf nodes. Since q is high for OSSPN and this indicates unseen values and overfitting, $\alpha = 0.4$ is chosen for OSSPN.

The evaluation of $\kappa_{\min}$ is shown in table 6.6. Here, $\kappa_{\min}$ is varied and relative values of 1 %, 0.5 % and 0.1 % of the training data are assessed. During the comparisons constant values $\alpha = 0.3$ and $\iota = 100\,000$ are used.

Table 6.6 Evaluation of results for the variation of $\kappa_{\min}$. $\kappa_{\min} = 0.1\%$ is chosen for MSPN_{RDC}. For the rest, it $\kappa_{\min} = 1\%$ selected. The table was published by Jesenski et al. [135]; © 2020 IEEE.

algorithm	$\kappa_{\min}$	$\mathfrak{Li}_{\log}$	q	E	t_{train} in s
LSPN	1%	−14.96	$\mathbf{3.88 \times 10^{-7}}$	**626.7**	**7688**
	0.5%	−14.93	1.17×10^{-6}	907.0	7957
	0.1%	**−14.91**	1.17×10^{-6}	1919.2	8401
MSPN_{km}	1%	−14.96	0	**626.7**	**7698**
	0.5%	−14.93	0	907.0	7950
	0.1%	**−14.91**	0	1919.2	8401
MSPN_{RDC}	1%	−15.36	0	**804.0**	**13 588**
	0.5%	−15.14	0	1314.8	14 206
	0.1%	**−14.78**	0	4034.5	16 144
SSPN	1%	−15.00	$\mathbf{3.88 \times 10^{-7}}$	**855.8**	**9664**
	0.5%	−14.97	1.17×10^{-6}	1277.0	10 060
	0.1%	**−14.95**	1.17×10^{-6}	2627.3	11 263
OSSPN	1%	−15.00	$\mathbf{5.23 \times 10^{-4}}$	**869.3**	**7041**
	0.5%	−14.96	6.75×10^{-4}	1341.5	11 184
	0.1%	**−14.94**	1.02×10^{-3}	2632.3	30 904

Table 6.7 Evaluation of the variation of ι. $\iota = 200\,000$ is selected for the work in the remainder. The table was published by Jesenski et al. [135]; © 2020 IEEE.

algorithm	ι	$\mathfrak{Li}_{\log}$	q	E	t_{train} in s
OSSPN	50 000	−14.99	1.47×10^{-3}	**885.8**	**6032**
	100 000	**−14.96**	$\mathbf{6.75 \times 10^{-4}}$	1341.5	11 904
	200 000	−14.97	7.73×10^{-4}	1392.7	11 921

Table 6.6 shows, that except for MSPN_{RDC} the influence of $\kappa_{\min}$ on the likelihood of the networks is rather small compared to the influence of α. Therefore, $\kappa_{\min} = 1\,\%$ is selected since this creates networks with a smaller number of edges which corresponds to shorter inference times. Since the accuracy of MSPN_{RDC} is strongly influenced by $\kappa_{\min}$, 0.1 % is chosen at this end despite the growing number of edges.

Using a proper value of ι is essential when training OSSPN. Therefore, networks are trained and assessed for the values $\{50\,000, 100\,000, 200\,000\}$. During the variation of ι, the other parameters are kept constant at $\alpha = 0.3$ and $\kappa_{\min} = 0.5\,\%$. The results are illustrated in table 6.7. A number of $\iota = 50\,000$ should not be selected since this causes a very high amount of impossible samples q. Since the training time and the number of edges E is nearly identical for 100 000 and 200 000, $\iota = 200\,000$ is preferred since that could be more robust. A restriction of this assessment is, that it is executed on a dataset which does not change strongly over time. In order to find good values for OSSPN, the assessment should be repeated on an evolving dataset which strongly changes its probability distribution over time.

6.3.3.2 Inter-algorithmic comparison

Table 6.8 shows the hyper-parameter combinations which were - as a result of the assessment in the previous section - chosen for an inter-algorithmic comparison. The results of the comparison are shown in table 6.9. The table shows, that q is very small for all algorithms except OSSPN. This can be explained by the fact that OSSPN only trains the network's parameters on ι recent samples.

MSPN_{RDC} creates the best likelihoods $\mathfrak{Li}_{\log}$ among the SPNs, however it requires comparably long sampling times. LSPN, SSPN and MSPN_{km} possess similar likelihoods, but MSPN_{km} needs the shortest sampling time

Table 6.8 Overview about the selected hyper-parameters. The table was published by Jesenski et al. [135]; © 2020 IEEE.

hyper - param.	LSPN	MSPN_{km}	MSPN_{RDC}	SSPN	OSSPN
κ_{min}	1%	1%	0.1%	1%	1%
α	0.3	0.3	0.3	0.3	0.4
Δ	-	1	1	-	-
$\mathcal{N}_0$	-	-	-	factorized product node	
ι	-	-	-	-	200 000
Λ	-	-	-	-	100 000

Table 6.9 Inter-algorithmic comparison of results. t_{sample} represents the mean time needed to sample a lane section of a length of 480 m. The sampling times were determined on an Intel Xeon E-2144G with 64 GB of RAM. The table was published by Jesenski et al. [135]; © 2020 IEEE.

algorithm	$\mathfrak{Li}_{\text{log}}$	q	t_{train} in s	t_{sample} in s
LSPN	-14.96	3.88×10^{-7}	7688	2.80
MSPN_{km}	-14.96	**0**	7698	**2.44**
MSPN_{RDC}	-14.78	**0**	16 144	12.37
SSPN	-15.00	3.88×10^{-7}	9664	4.11
OSSPN	-15.24	3.46×10^{-4}	**6052**	2.85
BN	**-13.79**	**0**	54 122	87.30

(2.44 s) for a whole statistically populated road with a length of more than 400 m. An example for a generated scene sampled by MSPN_{km} is given in fig. 6.25.

Compared to the BN approach, all SPNs show a worse average log-likelihood. However, the SPNs can sample whole lane sections up to 36 times faster: Whereas the BN needs about 87 s, the fastest SPN populates a lane section in 2.4 s. Additionally, the BN's training requires much more time than the SPNs' training.

Further comparisons between the algorithms can be performed on statistics of properties of completely filled lane sections. A few statistical evaluations of populated lane sections are given in figs. 6.26 and 6.27. These evaluations analyze 20 000 scenes generated by each of the SPN algorithms and compare them to results of the BN. Due to sampling time constraints

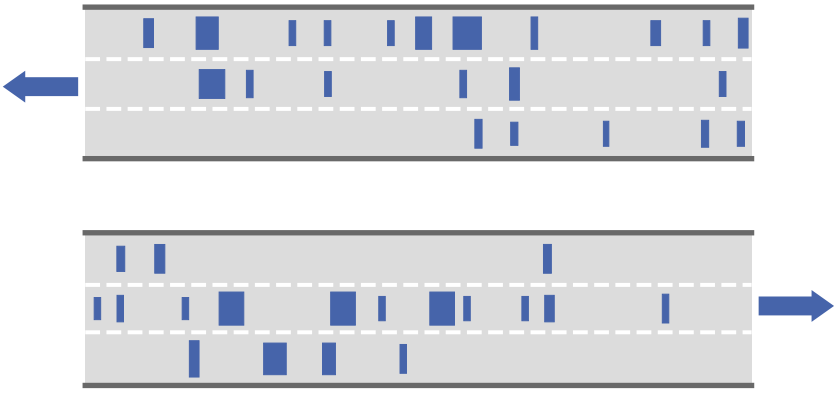
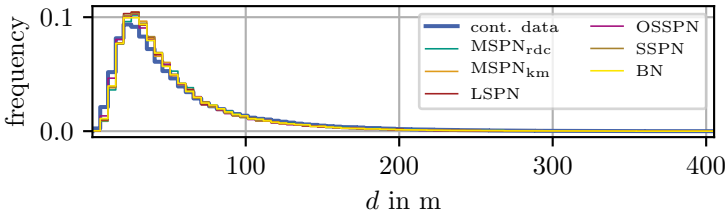
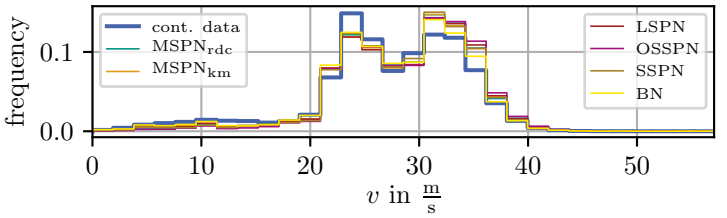


Figure 6.25 Scene generated by using SPNs which were trained by MSPN_{km}. The displayed 6-lane highway section has a length of about 480 m and a width of about 30 m. Hence, the vertical and horizontal scales in the image are strongly deviating and therefore the vehicles appear distorted.

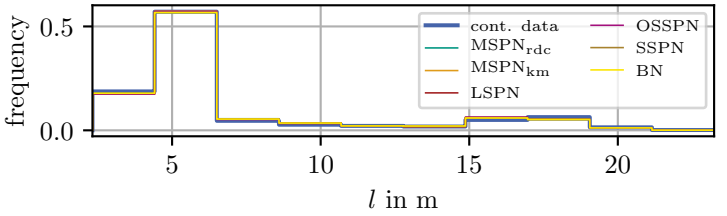
only 2000 scenes sampled by the BN were used for the statistics. Additionally, the statistics extracted from the scenes which were contained in the test dataset are given. Figures 6.26(a) to 6.26(c) show statistics of variables on which the networks were trained. Figures 6.27(a) and 6.27(b) illustrate emergent variables which were not directly trained. Naturally, the performance of the networks is better on the trained variables. Here, the performance of the different networks seems to be comparable. All networks can qualitatively reproduce the statistics of the learned variables in figs. 6.26(a) to 6.26(c). For fig. 6.27(a), there are deviations of invTTC values next to $0\,\text{s}^{-1}$. The displayed discretized test data show that this is completely caused by the discretization of the dataset. All networks overestimate the vehicle density as shown in fig. 6.27(b). Consistently, the networks' distance distributions in fig. 6.26(a) overestimate small distance values. The Bayesian network baseline seems to reproduce the vehicle density slightly better than the SPNs.



(a) Distance between consecutive vehicles.



(b) longitudinal velocity



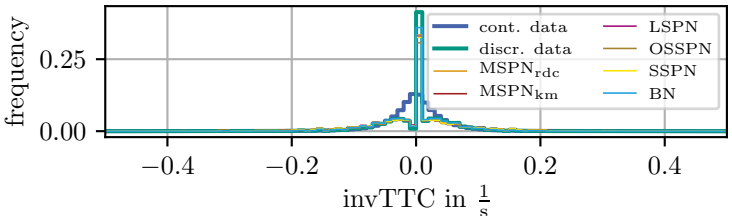
(c) length

Figure 6.26 Comparison of histograms of directly trained properties of populated lanes sampled by the SPNs with the continuous test data and the BN baseline. **(a)** shows the statistics of the longitudinal distance between vehicles. **(b)** displays the distribution of velocity. **(c)** contains information about the lengths of the sampled vehicles. The graphs were firstly published by Jesenski et al. [135]; © 2020 IEEE.

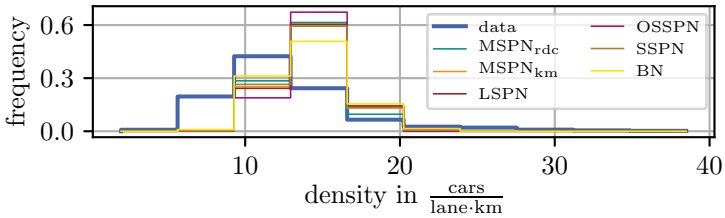
6.4 Discussion/summary

This chapter aims to solve open questions regarding the generation of scenes. In detail, the handled questions concern a scene model's applicability to complex road topologies and its efficiency when sampling.

For that, section 6.2 describes the implementation of a framework of BNs which allows modeling traffic scenes on complex intersections. The proposed



(a) invTTC



(b) number of vehicles

Figure 6.27 Comparison of distributions for emergent variables which were not directly trained in the networks. **(a)** shows a histogram of the inverse time to collision (invTTC) between consecutive vehicles and **(b)** shows the vehicle density of the populated lane sections. **(a)** also contains the statistics of the discretized test data. The histograms were firstly published by Jesenski et al. [135]; © 2020 IEEE.

model enables modeling inhomogeneous curved lane sections and several types of relations between them. Additionally, it includes the possibility to model global properties with intersectional influence. The model was illustrated by applying it to two intersections: the KoPER intersection in Aschaffenburg and the intersection at Neuköllner Straße, Aachen which is contained in the inD dataset. By using multiple metrics it could be shown that the implementation of the proposed model reproduces the datasets accordingly. By the application to the signalized KoPER intersection and the unsignalized inD intersection, it was shown that the model adapts to different types of road topologies. Eventually, section 6.2 shows the fulfillment of contribution 3.

Section 6.3 proposes the usage of SPNs to populate road topologies. Therefore by learning distributions from the highD dataset, four different SPN learning algorithms were compared to a BN baseline. While the SPNs sample a scene up to 36 times faster than the BN, they show a worse performance regarding the log likelihood. Nevertheless, it was shown that

the SPNs can qualitatively reproduce the dataset. Additionally, the short sampling time is essential for simulation-based validation methods (e.g. based on importance sampling; see chapter 5) since they usually require a high amount of sampled scenes. Among the compared SPN learning algorithms, MSPN_{rdc} performs best regarding the likelihood, but has the worst sampling time. LSPN, MSPN_{km} and SSPN generate similar likelihods. Among them, MSPN_{km} shows the shortest sampling time. OSSPN performs worst which is caused by its online setting. However, there might be some applications where the online property and the ability not to load the whole dataset at once could become valuable. In total, section 6.3 delivers all ingredients for the realization of contribution 4.

The high number of lane sections, which is necessary to describe complex road topologies, creates a large number of combinations which splits the available training data. This results in some parameter combinations being trained only on small data subsets. Hence, it would be beneficial to train the models on larger datasets. Additionally, larger datasets would be sensible in order to enable the inclusion of neglected vehicle properties such as lateral position or orientation angles relative to the road centerline into the trained distributions.

As mentioned in chapter 5, a proper IS optimization of high dimensional ADF-simulations demands the ability to optimize the initial scenes which start the simulation runs. By using the results of this chapter and by introducing some adaptions to the scene models, it will become possible to predominantly sample critical scenes in chapter 7. This will enable optimization and hence the incorporation into importance sampling. Hence, chapter 7 will illustrate that a combination of the scene models with IS is possible.

7 Combining importance sampling and scene generation models

Chapter 5 proposes a method to apply IS to highly complex (highway) scenarios. However, the initial scenes for the simulation runs were created by just drawing from a dataset. This made it impossible to optimize the initial scenes when trying to find an optimal IS distribution and necessitated the introduction of a relaxation phase (see section 5.3.3).

Of course, for the calculation of the collision rates of real systems, a relaxation phase must not be used since it introduces unrealistic dynamics and biases the statistical model. Hence, it must become possible to optimize the initial scenes when implementing IS. In order to address this issue, the PGM-based initial scene model discussed in chapter 6 shall now be adapted to being able to produce predominantly critical scenes. This is realized by introducing parameters which are linked to the criticality of an initial scene into the relevant sampling distributions.

In section 7.1 specific safety metrics are tested for this purpose. In section 7.2 it is shown that the adapted initial scene models can be used to include the initial scenes into the CE optimization and a good approximation of the optimal IS distribution can be found. This renders the relaxation phase unnecessary since all relevant parts of a simulation run now can be optimized. Parts of this chapter were already published by Jesenski et al. [136];

7.1 Generation of critical scenes

This section discusses how to adapt the initial scene model in order to generate critical scenes. Therefore, section 7.1.1 adjusts the scene model by adding additional metric parameters. Section 7.1.2 presents the preprocessing of the required dataset. Section 7.1.3 shows that discriminative learning

is advantageous when training the necessary probability distributions and section 7.1.4 discusses the effects of several metrics when using them to select a certain criticality during scene sampling.

7.1.1 Adaption of initial scene model

In section 6.1.1 it was described how to generate initial scenes by sampling the first vehicle in a lane $\mathcal{L}$ from a distribution $p_1^{(\mathcal{L})}$ (see (6.1)) and by iteratively sampling the following vehicles from a distribution $p_{\text{next}}^{(\mathcal{L})}$ (see(6.2)). However, this enabled only the reproduction of scene distributions as given in the original dataset.

In order to enable the generation of scenes with a distinct criticality, a set of random variables $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ which describe the safety metrics is now introduced into $p_{\text{next}}^{(\mathcal{L})}$. One obtains

$$\begin{aligned} p_{\text{next,cond.}}^{(\mathcal{L})} &\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}\right) \\ &= p_{\text{cond.}}^{(\mathcal{L})} \left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right) \\ &\cdot p_{\text{metrics}}^{(\mathcal{L})} \left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}\right), \end{aligned} \tag{7.1}$$

with

$$\boldsymbol{\Psi}_{i+1,\mathcal{L}} = \{\Delta v_{i+1}, d_{i+1}, l_{i+1}, w_{i+1}\}, \tag{7.2}$$

$$\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}} = \left\{v_i, v_{i-1}, d_i, d_{\text{right},i}^{(\text{ego})}, d_{\text{right},i}\right\}. \tag{7.3}$$

For the definition of the variables compare section 6.1.1. Remember that $\varphi_{\boldsymbol{X}}$ are numerically assigned values of the random variables in $\boldsymbol{X}$. The set $\boldsymbol{\Psi}_{i+1,\mathcal{L}}$ contains the random variables which are required to determine vehicle $i+1$ in lane $\mathcal{L}$ and $\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}$ gives the random variables of vehicle i in the same lane on which it is conditioned during the sampling of $\boldsymbol{\Psi}_{i+1,\mathcal{L}}$. In the following, both $p_{\text{cond.}}^{(\mathcal{L})}$ and $p_{\text{metrics}}^{(\mathcal{L})}$ are represented by SPNs learned on a dataset.

Note, that the sampling of the velocity v_{i+1} of the next vehicle was replaced by sampling the velocity difference $\Delta v_{i+1} = v_{i+1} - v_i$ since this is much more relevant for safety than v_{i+1} itself. Hence, directly training on Δv_{i+1} when learning $p_{\text{next,cond.}}^{(\mathcal{L})}$ improves the expressiveness of the used learning approaches with regard to a scene's safety.

The set of "metric parameters" $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ expresses safety in the relation between consecutive vehicles i and $i+1$ in lane $\mathcal{L}$. Note, that $\boldsymbol{\Psi}_{i+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ makes sure that no parameter is sampled twice if some of the used metrics are also included in $\boldsymbol{\Psi}_{i+1,\mathcal{L}}$. Depending on the sampling of $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ in the second factor $p^{(\mathcal{L})}_{\text{metrics}}$, the safety/criticality of the generated scenes can be influenced. Nevertheless, since the distribution $p^{(\mathcal{L})}_{\text{metrics}}\left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}\right)$ is learned from data (e.g. by SPNs or BNs as proposed in chapter 5), the entire distribution $p^{(\mathcal{L})}_{\text{next,cond.}}$ still represents the same unskewed distribution as $p^{(\mathcal{L})}_{\text{next}}$.

However, when trying to sample skewed, predominantly critical scenes it is possible to replace $p^{(\mathcal{L})}_{\text{metrics}}$ and sample the following vehicles of lane $\mathcal{L}$ by

$$\begin{aligned} q^{(\mathcal{L})}_{\text{next,cond.}}&\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}}\setminus\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}, \varphi_{\mathfrak{v}_{\text{init},\mathcal{L},i}}\right) \\ &= p^{(\mathcal{L})}_{\text{cond.}}\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}}\setminus\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right) \\ &\cdot q^{(\mathcal{L})}_{\text{metrics}}\left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\mathfrak{v}_{\text{init},\mathcal{L},i}}\right). \end{aligned} \tag{7.4}$$

Hence, by the use of the parametric family of distributions $q^{(\mathcal{L})}_{\text{metrics}}$ which can be chosen to emphasize certain ranges of values of the metric parameters included in $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$, it becomes possible to sample predominantly from the parts of the learned dataset which relate to a certain degree of criticality. The parameters $\mathfrak{v}_{\text{init},\mathcal{L},i}$ shall fully specify the distribution $q^{(\mathcal{L})}_{\text{metrics}}$.

7.1.2 Dataset & data preprocessing

For the SPN-based training of the modified initial scene models (7.1) and (7.4), the highD dataset (appendix B.2), which is already used in section 6.3, is utilized. Therefore, the dataset preprocessing which was explained in section 6.3.1 is also applied here before the training of the modified scene models takes place. However, since the metric parameters contained in $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ must additionally be considered, further preprocessing steps are executed.

Most of the metrics tested and compared for $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ (see section 7.1.4) are not included in the original data. Therefore, these metrics must firstly be calculated from the available dataset parameters in order to enable learning them. The metrics, which are calculated, are $g^{(\text{TTC})}_{\text{pair}}$, $g^{(\text{invTTC})}_{\text{pair}}$ and

the physics-based $g_{\text{pair}}^{(\text{RSS Dist})}$ metric. For more information on these metrics see chapter 4. They are applied on consecutive vehicle pairs $p = (i, i+1)$. For the purpose of $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$, only the longitudinal part of $g_{\text{pair}}^{(\text{RSS Dist})}$ was used since this is the prevailing metric term for consecutive vehicles in the same lane (and the SPN fills a lane consecutively). For the parameterization of $g_{\text{pair}}^{(\text{RSS Dist})}$, $\rho = 0.5\,\text{s}$, $a_{\text{max,accel}} = 4\,\frac{\text{m}}{\text{s}^2}$, $a_{\text{min,brake}} = 4\,\frac{\text{m}}{\text{s}^2}$ and $a_{\text{max,brake}} = 4\,\frac{\text{m}}{\text{s}^2}$ were used. In section 7.1.4, the three mentioned metrics additionally will be compared to a 2-dimensional metric $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}} = \{d_{i+1}, \Delta v_{i+1}\}$, i.e. the criticality is in this case directly determined by setting the distance and velocity difference to the vehicle which is sampled next. The values for d_{i+1} as well as for Δv_{i+1} are already included in the dataset and therefore no new calculations are necessary for this metric selection.

The metrics TTC, invTTC, as well as RSS tend to produce diverging values since they include a quotient and the denominator is able to reach 0. These possible diverging or at least very large values must be handled before the networks' training can be started. For RSS, all infinite values are replaced with the highest finite RSS value found in the dataset. For TTC and invTTC it becomes more complicated since all negative values are uncritical. Therefore, for TTC all values not included in the interval $[0\,\text{s}, \infty)$ are set to the highest available finite value. For invTTC the highest values are the most critical ones. Hence, the procedure must be adapted. That means, all negative invTTC values are set to the smallest positive value found in the dataset. All positive, infinite values are set to the highest finite value occurring in the dataset.

As already explained in section 6.3.1, the data are discretized before using them to train the SPNs. Up to now, equidistantly distributed bins were used. However, the metric values are distributed strongly uneven in a very large range, especially because of their diverging behavior. Consequently, equidistant bins are not feasible anymore and the metric bins are arranged to make sure that each metric bin contains the same quantile of metric data[1]. For $g_{\text{pair}}^{(\text{TTC})}$, $g_{\text{pair}}^{(\text{invTTC})}$ and $g_{\text{pair}}^{(\text{RSS Dist})}$, 10 of those quantile-based bins per metric are used. When handling negative metric values or infinite values as described above, a lot of metric values might be placed at the exactly same numerical value. In these cases, potentially multiple quantile-based bins are located within this single value. If that happens, the overlapping bins

[1] The rest of the SPN parameters $\boldsymbol{\Psi}_{i+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}, \boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}$ are of course still discretized equidistantly.

however are fused together. For the metric $\{d_{i+1}, \Delta v_{i+1}\}$, d_{i+1} is discretized quantile-based and Δv_{i+1} is discretized equidistantly. The reason is that the networks will be trained discriminatively for d_{i+1} and generatively for Δv_{i+1}. Section 7.1.3 discusses such generative and discriminative training procedures. A number of 80 bins were used for discretizing d_{i+1} and 30 were applied to discretize Δv_{i+1} when using them as metrics in $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$.

7.1.3 Discriminative learning of $p^{(\mathcal{L})}_{\text{cond.}}$

This section discusses how to train SPNs in order to properly represent the conditional distribution

$$p^{(\mathcal{L})}_{\text{cond.}}\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}}\setminus\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right). \tag{7.5}$$

Usually, when training an SPN, the joint distribution

$$p^{(\mathcal{L})}_{\text{cond.}}\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}}\setminus\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}, \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right) \tag{7.6}$$

is learned generatively. The conditioning on $\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}, \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ is then obtained by dividing two marginalizations as described in eq. (6.23). However, better results can be achieved when directly learning the desired conditional distribution. Consequently, the discriminative training of a conditional distribution

$$p^{(\mathcal{L})}_{\text{cond.}}\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}}\setminus\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}, \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right) \tag{7.7}$$

is beneficial[2]. This allows improving the learning of the relations between the consecutive vehicles for a distinct value of the respective metric. The preprocessed training dataset can be visualized as sequence of data tuples with values for all random variables in $\boldsymbol{\Psi}_{i+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$, $\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}$ and $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$. The direct conditional training can be implemented by dividing this total dataset into subsets whose data tuples belong to a certain bin $\mathfrak{b}$ of the metric in $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$. Then a separate $\text{SPN}_{\mathfrak{b}}$ can be trained on each of these sub-datasets, which means that in effect the training is executed in a discriminative fashion. The conditioning on a distinct value for $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ is then achieved by selecting the SPN from the set $\{SPN_{\mathfrak{b}} : \mathfrak{b} \in \{0, 1, ...\}\}$

[2] The desired distribution in (7.5) can then be obtained by the scheme $p\,(a|b,c) = \frac{p(a,b|c)}{p(b|c)}$

which corresponds to the respective desired metric bin[3]. In section 6.3.3.2, it was discussed that MSPN_{km} delivers the best results among the analyzed training algorithms. It is thus used to train the SPNs in this section.

The quality of the learned discriminative distributions with respect to their ability to represent the conditional influence of the metric in distribution (7.5) can be evaluated by sampling from the marginalized distribution

$$p_{\text{cond.}}^{(\mathcal{L})}\left(\varphi_{\Delta v_{i+1}}, \varphi_{d_{i+1}}, \varphi_{v_i} \middle| \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right). \tag{7.8}$$

For doing so, the discriminative SPNs which represent (7.7) were marginalized and the sampling is hence limited to $\Delta v_{i+1}, d_{i+1}, v_i$. This is sensible since only these parameters influence the tested metrics used in $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$. The rest of the parameters are therefore not relevant for the evaluation and are neglected at this point. The sampled parameters $\Delta v_{i+1}, d_{i+1}, v_i$ can then be used to recalculate the metrics in $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ on which the networks were conditioned. If these recalculated metrics are distributed as desired by the given value of $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$, then a good discriminative representation of the conditional distribution (7.5) was achieved. As an example, histograms of the recalculated values of $g_{\text{pair}}^{(\text{TTC})}$ are given in fig. 7.1. The sub-figures compare the results for the recalculated metric values when the SPN was naively trained in a generative way and the results when the SPNs were trained discriminatively with the mentioned procedure. The figure shows that the discriminative approach creates a better representation of (7.5) because it produces larger shifts between the recalculated metric values than the generative approach. Consequently, in the remainder of this work, the discriminative learning approach will be applied to learn $p_{\text{cond.}}^{(\mathcal{L})}$.

[3] This learning procedure was chosen since it allows using the generative learning approaches presented in section 6.1.3 in order to train discriminative networks. In literature (e.g. [5, 84]), a similar SPN structure was utilized in order to train classifiers of type $p(Y|\boldsymbol{X})$. Here Y is a number representing the class related to the features in the set $\boldsymbol{X}$. The number of possible values of Y is low, the number of values for $\boldsymbol{X}$ is high (scales exponentially with $|\boldsymbol{X}|$). In the present case this is inverted to $p(\boldsymbol{X}|Y)$ and therefore trivial to represent since only few networks for the few values of Y must be trained.

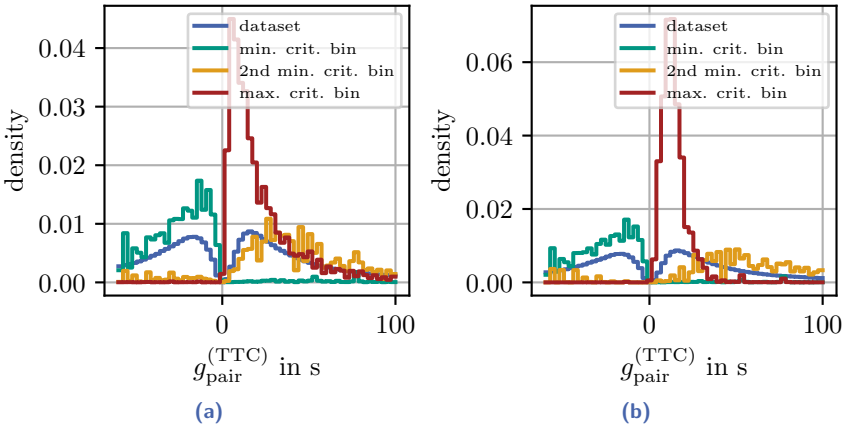


Figure 7.1 Histograms of recalculated metric values. In both histograms the sampling was conditioned on the most critical metric bin (*max. crit. bin*), on the minimal critical bin (*min. crit. bin*) and on the second least critical bin (*2nd min. crit. bin*) for $\phi_{\text{metrics},i,\mathcal{L}} = \left\{ g_{\text{pair}}^{(\text{TTC})} \right\}$. The results are shifted consistently into the direction of the conditioned bins since higher values for $g_{\text{pair}}^{(\text{TTC})}$ are less critical. Remember that negative TTC values are absolutely uncritical since they imply a situation with a faster leading vehicle. For comparison, the distribution of the TTC values in the dataset is also included. Note that the plotted dataset distribution is taken from the original data prior to the application of the processing steps described in section 7.1.2. Notice that the min. crit. bin corresponds to the bin with the highest TTC values. However, conditioning onto this highest bin creates a lot of negative recalculated TTC values. This is caused by the preprocessing introduced in section 7.1.2: Before training, all negative values are set to the highest finite TTC value found in the dataset. Therefore, these data tuples now are related to the least critical metric bin (with the highest value) during training. The networks were trained by MSPN_{km} because this training method delivered the best overall results as analyzed in section 6.3.3.2. The histograms were produced by evaluating 10 000 samples per metric bin drawn by (7.8). **(a)** shows the histogram of TTC as sampled by one SPN for all conditioned values of $\phi_{\text{metrics},i,\mathcal{L}} = \left\{ g_{\text{pair}}^{(\text{TTC})} \right\}$ and by dividing by the marginalizations (generative learning, based on (7.6)). **(b)** gives the same histogram of TTC when sampling with an extra $\text{SPN}_{\mathfrak{b}}$ per discretized TTC bin $\mathfrak{b}$ (discriminative learning, based on (7.7)). The results of **(b)** are superior since a better separation of TTC values takes place. That means, 2nd min. crit. bin for example is located at higher, less critical TTC values than in **(a)** and max. crit. bin is also more pronounced on smaller, more critical values. Figure 7.1(b) was firstly published by Jesenski et al. [136] © 2021 IEEE.

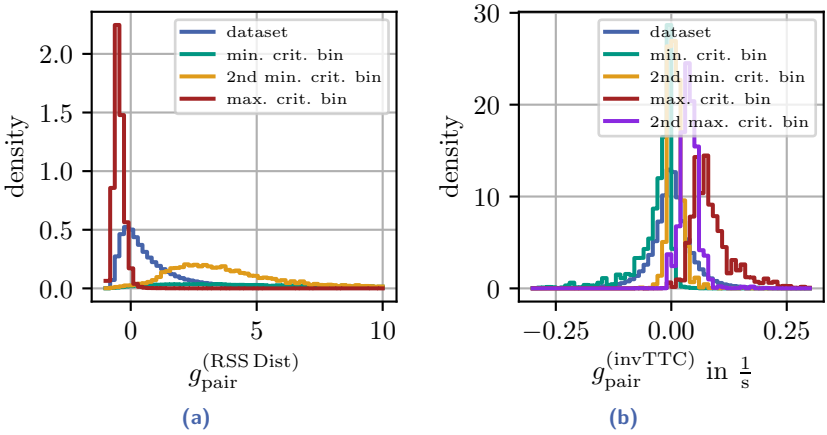
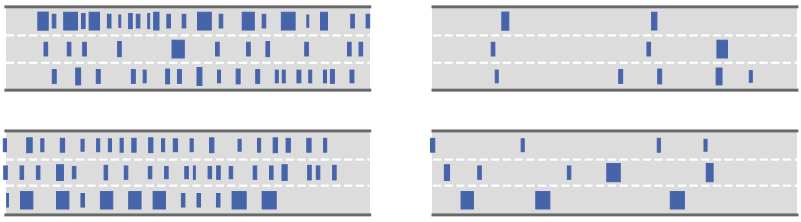


Figure 7.2 Histograms for recalculated metric values for both $g_{\text{pair}}^{(\text{invTTC})}$ and $g_{\text{pair}}^{(\text{RSS Dist})}$. Compare also fig. 7.1(b). The histograms were constructed based on 10 000 samples per metric bin setting. **(a)** gives the histogram of RSS sampled with an extra SPN per discretized RSS bin. The conditioning is working. Most of the values sampled when conditioning on the maximal critical bin are located in the metric's critical range $[-1, 0]$ (see section 4.4.2), whereas the other uncritical conditioned values are located at high recalculated metric values. **(b)** displays the same results for invTTC. The conditioning is also working here, since the more critical the conditioned bins are, the higher also the recalculated invTTC values are. Note again, that the minimal critical bin delivers a lot of uncritical negative invTTC values since all negative values were positioned at the lowest positive invTTC value as a result of the procedure explained in section 7.1.2. Figure 7.2(a) was firstly published by Jesenski et al. [136] © 2021 IEEE.

7.1.4 Comparison of different metrics

In addition to sampling critical scenes by $\phi_{\text{metrics},i,\mathcal{L}} = \left\{ g_{\text{pair}}^{(\text{TTC})} \right\}$ as presented in section 7.1.3, the sampling by $g_{\text{pair}}^{(\text{invTTC})}$ and $g_{\text{pair}}^{(\text{RSS Dist})}$ was evaluated. The evaluation was performed similar to the assessment of $g_{\text{pair}}^{(\text{TTC})}$ in fig. 7.1. The SPNs for both metrics were learned by applying MSPN_{km} through the discriminative approach. The results are displayed by fig. 7.2. Obviously, the conditioning works for both metrics.

In order to compare the effects of the conditioning on the various metrics to each other, entire scenes are sampled by the iterative application of (7.4). For that, the required factor $p_{\text{cond.}}^{(\mathcal{L})}$ is described by the respective discriminatively trained distribution. This was executed with



(a) (b)

Figure 7.3 Scenes of varying criticality generated by directly setting the criticality for all sampled vehicles. The displayed 6-lane highway section has a length of about 480 m and a width of about 30 m. Hence, the vertical and horizontal scales in the images are strongly different and the vehicles appear distorted. For (a), it is $q^{(\mathcal{L})}_{\text{metrics}}\left(\varphi_{g^{(\text{RSS Dist})}_{\text{pair}}}\right) = \delta_{\varphi_{g^{(\text{RSS Dist})}_{\text{pair}}},\text{max.crit.bin}}$. (b) was drawn by $q^{(\mathcal{L})}_{\text{metrics}}\left(\varphi_{g^{(\text{RSS Dist})}_{\text{pair}}}\right) = \delta_{\varphi_{g^{(\text{RSS Dist})}_{\text{pair}}},\text{min.crit.bin}}$. Both images were already published by Jesenski et al. [136] © 2021 IEEE.

$q^{(\mathcal{L})}_{\text{metrics}}\left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\mathbf{v}_{\text{init},\mathcal{L},i}}\right)$ set to sample only from the minimal critical bin (min. crit. bin) or the maximal critical bin (max. crit. bin) of the metric, respectively[4]. Exemplary results of single scenes drawn with $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}} = \left\{g^{(\text{RSS Dist})}_{\text{pair}}\right\}$ are given in fig. 7.3. Obviously, the effects of conditioning on the metric are significant in these examples. The scene which was sampled by conditioning on max. crit. bin contains a lot more vehicles with much smaller distances d between consecutive vehicles in a lane than the scene sampled with min. crit. bin.

Now, the scene creation procedure was applied to create 2000 complete scenes for both of the mentioned bins. For comparison, the same amount of scenes was also drawn with $p^{(\mathcal{L})}_{\text{metrics}}$ as defined in (7.1). In this relation, $p^{(\mathcal{L})}_{\text{metrics}}$ was trained by MSPN_{km} to represent the real distribution as included in the dataset.

The safety with regard to two consecutive vehicles in a scene is predominantly dependent on their velocity difference Δv and their distance d. Thus, the histograms of Δv and d within the 2000 generated entire scene samples

[4] Because of the discretization of the metric, $q^{(\mathcal{L})}_{\text{metrics}}$ is of course a discrete probability mass function which in this case is set to be defined by a Kronecker Delta.

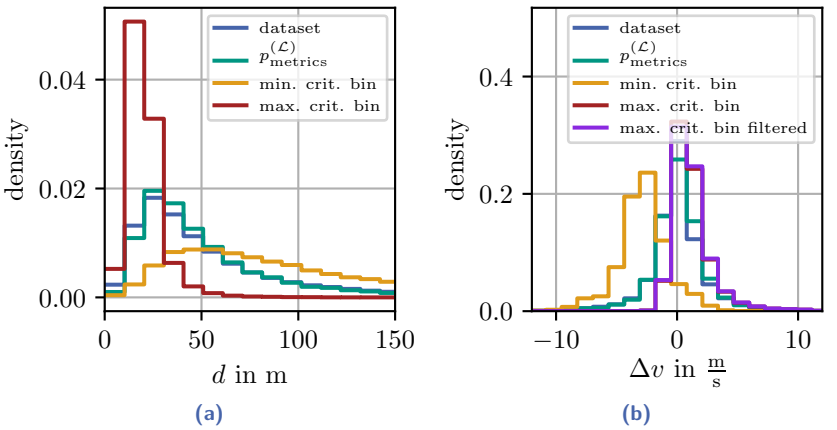


Figure 7.4 Histograms for d and Δv taken from 2000 scenes per metric bin generated with $\phi_{\text{metrics},i,\mathcal{L}} = \left\{ g^{(\text{RSS Dist})}_{\text{pair}} \right\}$. They are compared with the respective histograms of the highD dataset. The images show, that when sampling from the SPN $p^{(\mathcal{L})}_{\text{metrics}}$, the dataset distributions are approximately reproduced. The rest of the histograms were sampled by $q^{(\mathcal{L})}_{\text{metrics}}$, e.g. for min. crit. bin, it is sampled by $q^{(\mathcal{L})}_{\text{metrics}}\left(\varphi_{g^{(\text{RSS Dist})}_{\text{pair}}}\right) = \delta_{\varphi_{g^{(\text{RSS Dist})}_{\text{pair}}},\text{min.crit.bin}}$. **(a)** shows that the distances d are strongly shifted when conditioning on metric bins of varying criticality. **(b)** shows that the shift of Δv works for the uncritical bins, but the shift to more critical, higher values of Δv is only weak. For max. crit. bin filtered, values of $\Delta v = 0\,\frac{\text{m}}{\text{s}}$ are filtered out. Both images were already published by Jesenski et al. [136] © 2021 IEEE.

per bin have been analyzed and compared to each other. The histogram results for the three examined metrics are given in figs. 7.4 to 7.6. Before sketching the figures, the histograms were limited to interesting ranges in the x-direction. For Δv additionally a filtered version of max. crit. bin is drawn. For this histogram, values of $\Delta v = 0\,\frac{\text{m}}{\text{s}}$ are filtered. This becomes necessary, since the velocity in the sampled scenes is limited to $200\,\frac{\text{km}}{\text{h}}$. That means, if a lot of vehicles are located in a scene (which is the case for critically conditioned scenes) and the successor vehicles are driving faster ($\Delta v > 0$) this limit is reached pretty fast. Afterwards, the next vehicle is not allowed to drive faster even if a positive velocity difference is sampled. As a result, velocity differences of $0\,\frac{\text{m}}{\text{s}}$ are added and the histograms are skewed. When these velocity differences are filtered out, it becomes visible which velocity differences are chosen if a velocity increase is still allowed.

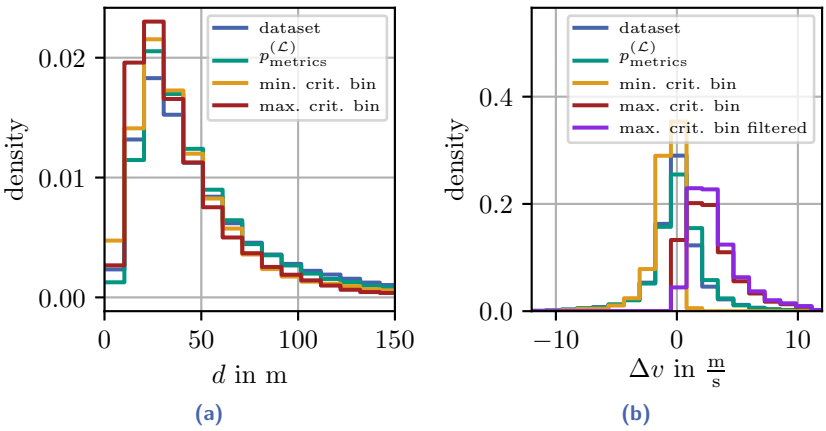


Figure 7.5 Histograms for d and Δv taken from 2000 scenes generated with $\phi_{\text{metrics},i,\mathcal{L}} = \left\{g^{(\text{TTC})}_{\text{pair}}\right\}$. The histograms are compared with the histograms of the highD dataset. **(a)** shows that TTC's influence on d is only of limited extent. However, **(b)** shows, that the shift induced into Δv when selecting highly critical metric bins is quite strong. The same filtered histogram as explained in fig. 7.4 is presented in **(b)**. Both images were already published by Jesenski et al. [136] © 2021 IEEE.

In the current cases, no strong skews are found by the filtering. The histograms are additionally compared to the histograms of the parameters in the original dataset. When comparing the figures, one can observe that the conditioning on the extremal critical values of $g^{(\text{TTC})}_{\text{pair}}$ and $g^{(\text{invTTC})}_{\text{pair}}$, introduces strong shifts in the histograms of Δv. However, the metrics' influence on d seems to be limited. In contrast, when utilizing $g^{(\text{RSS Dist})}_{\text{pair}}$, a strong shift to smaller values of d is visible, when it is conditioned on highly critical bins (fig. 7.4(a)). The influence on Δv (fig. 7.4(b)) however is weaker. The images show, that the sampling by all three metrics reproduces the dataset histograms when $p^{(\mathcal{L})}_{\text{metrics}}$ is applied for the sampling of the metric values. For all metrics, there are small deviations between the histogram sampled by $p^{(\mathcal{L})}_{\text{metrics}}$ and the histogram of the dataset at low values for d. These deviations seem to be similar to the deviations in fig. 6.26(a) and are therefore not additionally created by introducing the conditioning.

Consequently, the described conditioning procedure makes it possible to sample scenes of a certain criticality as quantified by different types of metrics. However, as the analysis reveals, none of the metrics alone seems

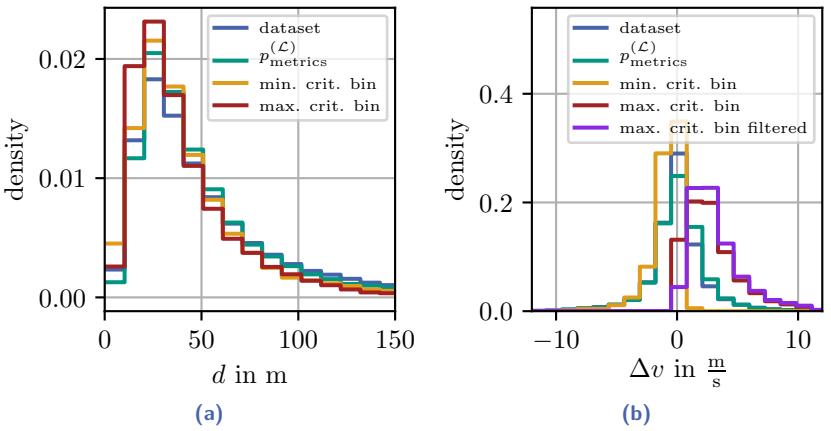


Figure 7.6 Histograms for d and Δv taken from 2000 scenes generated with $\phi_{\text{metrics},i,\mathcal{L}} = \left\{g^{(\text{invTTC})}_{\text{pair}}\right\}$. They are compared with the respective histograms of the highD dataset. The results are similar to the results in fig. 7.5. This is a result of the fact, that the values of invTTC are closely related to the values of TTC.

to be sufficient to fully enable the exhaustive optimization of all critical parts of an initial scene (at least $\Delta v, d$). As the experiments have shown, the conditioning on the tested metrics either allows tuning Δv or d, but not both combined[5]. Nevertheless, this is a requirement of the IS approach since the optimal importance sampling distribution (5.12) must weight up/emphasize all possible critical scenes. Too large deviations from this optimal importance sampling distribution after the optimization can cause faults in the calculation of the occurrence rates of a critical event. A solution is the direct conditioning on $\phi_{\text{metrics},i,\mathcal{L}} = \{d_{i+1}, \Delta v_{i+1}\}$. Obviously, this direct conditioning should enable the direct sampling of parameter values in the relevant critical ranges. The resulting histograms are illustrated in fig. 7.7. They show that it becomes possible to sample from the desired critical parameter ranges for both d and Δv. In order to keep the number of networks small (in this case 80), the discriminate learning has only been applied on d. However, there is no principal limitation preventing the

[5] In addition, when testing these metrics in the IS experiments described in 7.2 it turned out, that accidents can also happen at very uncritical values of these metrics, e.g. there may be situations where critical values of Δv_{i+1} (large) or critical values of d_{i+1} (small) evaluate to strongly uncritical values of RSS.

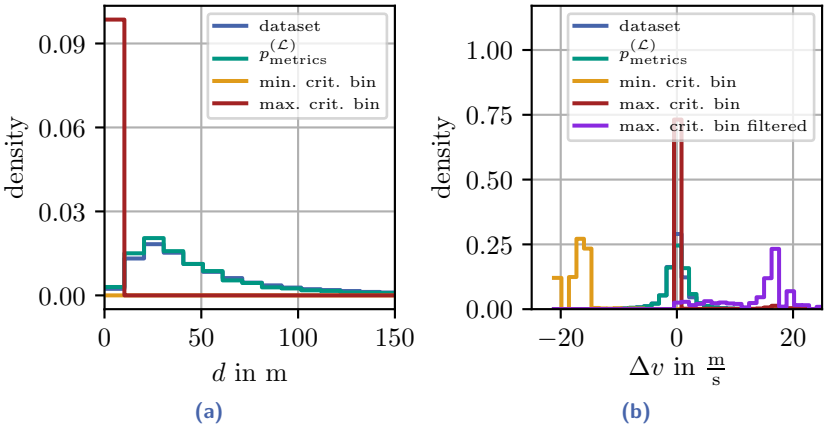


Figure 7.7 Histograms of Δv and d when sampling 2000 scenes by directly conditioning on $\phi_{\text{metrics},i,\mathcal{L}} = \{d_{i+1}, \Delta v_{i+1}\}$. When sampling by $p^{(\mathcal{L})}_{\text{metrics}}$, the dataset is reproduced properly. Sub-figure (a) shows that the generated values for d are located in a very narrow interval when sampling critically. This is sensible since these values were now directly sampled by choosing the most critical bin with the smallest available value for d when applying $q^{(\mathcal{L})}_{\text{metrics}}$. The respective sampled interval for min. crit. bin is not visible within the illustrated range of d since the minimal critical values for d correspond to the largest distance values found in the dataset, which are larger than 150 m. In (b), it is illustrated that the extremal bins for Δv are located at about $\pm 20\,\frac{\text{m}}{\text{s}}$. The unfiltered version of max. crit. bin however naively seems not being working properly since a large peak is visible around $0\,\frac{\text{m}}{\text{s}}$. Note however, that after having filtered $0\,\frac{\text{m}}{\text{s}}$, the Δv histogram looks as expected. The reason is that max. crit. bin samples very small distances d which causes a lot of vehicles being placed into a scene. Additionally, all of the consecutive vehicles shall maximally drive faster than their predecessor. This causes the maximum velocity of $200\,\frac{\text{km}}{\text{h}}$ allowed in the scenes being reached pretty fast and consequently a lot of successor vehicles with a velocity difference of $0\,\frac{\text{m}}{\text{s}}$ are following up. Note that both Δv peaks are distributed in a certain range and have a certain width, even for the filtered version. They are not only located in one bin as naively expected. There are two reasons for that: Firstly, as already introduced in chapter 6, for each of the lanes in the scene extra SPNs are trained. Therefore, different extremal bin values exist for each of those networks. Secondly, the limitation of allowed vehicle velocities to the interval $\left[0, 200\,\frac{\text{km}}{\text{h}}\right]$ causes larger/smaller values to occur in order to comply with these limits when the limits are approached. Both images were already published by Jesenski et al. [136] © 2021 IEEE.

discriminative learning scheme to be applied on both metrics.

Note, that even if a very specific metric set $\phi_{\text{metrics},i,\mathcal{L}} = \{d_{i+1}, \Delta v_{i+1}\}$ was chosen, the procedure is not limited. If IS further required condition-

ing on additional parameters (in $\boldsymbol{\Psi}_{i+1,\mathcal{L}}$) they can simply be added to $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$. Remember, that added parameters are deleted from $\boldsymbol{\Psi}_{i+1,\mathcal{L}}$. This is necessary, since it does not make sense to sample parameters twice. The parameters which should be added to $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}$ might for example be determined by methods similar to those in section 5.4.

7.2 Application to importance sampling

In chapter 5, a simulation framework was presented which was used to generate entire scenarios. The initial scenes were taken from the highD dataset and the vehicle dynamics were determined by the use of IDM and MOBIL. Now, this framework is adapted to use the modified initial scene model from section 7.1 which then enables including the initial scenes into the CE optimization. For that, section 7.2.1 discusses how the modified initial scene models are incorporated into the mentioned simulation framework. Section 7.2.2 presents the causality groups to perform IS and section 7.2.3 executes IS and shows the results.

7.2.1 Simulation model

The modified initial scene model in (7.1) is now introduced to the simulation framework from chapter 5. In that relation, a simulation run h is governed by (5.18). However, the non-physical parameters are now defined by

$$\boldsymbol{\phi} = \boldsymbol{\phi}_{\text{beh.}} \cup \boldsymbol{\phi}_{\text{scene}} \tag{7.9}$$

with the union

$$\boldsymbol{\phi}_{\text{scene}} = \bigcup_{\mathcal{L} \in \boldsymbol{L}} \bigcup_{i \in \boldsymbol{\nu}_{\varphi_\xi}^{(\mathcal{L})}} \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}} \tag{7.10}$$

of the metric parameters specified in (7.1). The set $\boldsymbol{\phi}_{\text{beh.}}$ contains the behavior models' parameters. They remain the same as discussed in section 5.3 since the dynamics of a simulation run shall still be governed by the behavior models given in section 5.3.1.3. The set $\boldsymbol{L}$ comprises all lanes existent at the current road topology and $\boldsymbol{\nu}_{\varphi_\xi}^{(\mathcal{L})}$ gives all vehicles located in lane $\mathcal{L}$.

When representing the unskewed creation of a simulation run h, the distribution $p_0\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}}\right)$ in (5.20) does no longer contain a factor $p_{0,\text{irr.},1}$,

which purely samples from a dataset, as it was the case in section 5.3.1.1. Instead, the initial scenes are sampled as discussed in sections 6.1.1 and 6.3 with the modifications in (7.1). Hence, the factors required in (5.20) are now defined by

$$\begin{aligned} p_{0,\text{irr.}}\left(\varphi_{\boldsymbol{x}_0}, \varphi_{\boldsymbol{\phi}_{\text{irr.}}} \middle| \varphi_{\boldsymbol{\phi}_{\text{scene}} \cap \boldsymbol{\phi}_{\text{rel.}}}\right) &= p_{0,\text{irr.}}^{(\text{beh.})}\left(\varphi_{\boldsymbol{\phi}_{\text{irr.}} \cap \boldsymbol{\phi}_{\text{beh.}}}\right) \\ &\cdot \prod_{\mathcal{L} \in \boldsymbol{L}} p_1^{(\mathcal{L})}\left(\varphi_{v_1}, \varphi_{d_\text{b}}, \varphi_{l_1}, \varphi_{w_1}\right) \\ &\cdot \prod_{\mathcal{L} \in \boldsymbol{L}} \prod_{i=1}^{n_{h,\mathcal{L}}-1} p_{\text{cond.}}^{(\mathcal{L})}\left(\varphi_{\boldsymbol{\Psi}_{i+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\right) \\ &\cdot \prod_{\mathcal{L} \in \boldsymbol{L}} \prod_{i \in \boldsymbol{\nu}_{\varphi_\xi}^{(\text{irr.},\mathcal{L})}} p_{\text{metrics}}^{(\mathcal{L})}\left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}\right) \\ &\cdot \prod_{\mathcal{L} \in \boldsymbol{L}} p_{\text{end}}^{(\mathcal{L})}\left(\varphi_{x_{n_{h,\mathcal{L}}+1}} < x_{\text{rear}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},n_{h,\mathcal{L}},\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},n_{h,\mathcal{L}},\mathcal{L}}}\right) \end{aligned} \tag{7.11}$$

and

$$\begin{aligned} p_{0,\text{rel.}}\left(\varphi_{\boldsymbol{\phi}_{\text{rel.}}}\right) &= p_{0,\text{rel.}}^{(\text{beh.})}\left(\varphi_{\boldsymbol{\phi}_{\text{rel.}} \cap \boldsymbol{\phi}_{\text{beh.}}}\right) \\ &\cdot \prod_{\mathcal{L} \in \boldsymbol{L}} \prod_{i \in \boldsymbol{\nu}_{\varphi_\xi}^{(\text{rel.},\mathcal{L})}} p_{\text{metrics}}^{(\mathcal{L})}\left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}\right). \end{aligned} \tag{7.12}$$

The distribution $p_{\text{cond.}}^{(\mathcal{L})}$ is created by the discriminative approach specified in section 7.1.3. The distributions $p_{\text{cond.}}^{(\mathcal{L})}$, $p_{\text{metrics}}^{(\mathcal{L})}$ and $p_1^{(\mathcal{L})}$ will in the remainder be represented by SPNs learned by MSPN_{km} which in section 6.3.3.2 delivered the best combination of sampling time and likelihood. Since the behavior models applied in section 5.3 are reused, the distributions $p_{0,\text{irr.}}^{(\text{beh.})}$ and $p_{0,\text{rel.}}^{(\text{beh.})}$ which consider the behavior parameters are chosen similar to the definition of $p_{0,\text{irr.},2}$ and $p_{0,\text{rel.}}$ in section 5.3.1.1. Furthermore, $\boldsymbol{\nu}_{\varphi_\xi}^{(\text{rel.},\mathcal{L})}$ and $\boldsymbol{\nu}_{\varphi_\xi}^{(\text{irr.},\mathcal{L})}$ give the vehicles whose initial states are relevant and not relevant respectively for the optimization of the initial scene. The number $n_{h,\mathcal{L}}$ gives the amount of vehicles initially positioned in lane $\mathcal{L}$ for simulation run h. The possibility that a hypothetical vehicle $n_{h,\mathcal{L}} + 1$ is located behind

the sampled road topology is determined by $p_{\text{end}}^{(\mathcal{L})}$. This distribution is a direct result of aborting the sampling if $\varphi_{d_{n_{h,\mathcal{L}}+1}} > \varphi_{x_{n_{h,\mathcal{L}}}} - \varphi_{l_{n_{h,\mathcal{L}}}} - x_{\text{rear}}$ Formally, it is therefore given by

$$\begin{aligned} p_{\text{end}}^{(\mathcal{L})} &\left(\varphi_{x_{n_{h,\mathcal{L}}+1}} < x_{\text{rear}} \middle| \varphi_{\boldsymbol{\Psi}_{\text{cond.},n_{h,\mathcal{L}},\mathcal{L}}}, \varphi_{\boldsymbol{\phi}_{\text{metrics},n_{h,\mathcal{L}},\mathcal{L}}} \right) \\ &= \int_{\boldsymbol{V}_{n_{h,\mathcal{L}}+1,\mathcal{L}}} p_{\text{cond.}}^{(\mathcal{L})} \left(\varphi_{\boldsymbol{A}_{n_{h,\mathcal{L}}+1,\mathcal{L}}} \middle| \varphi_{\boldsymbol{B}_{\text{cond.},n_{h,\mathcal{L}},\mathcal{L}}} \right) \mathrm{d}\varphi_{\boldsymbol{A}_{n_{h,\mathcal{L}}+1,\mathcal{L}}}, \end{aligned} \tag{7.13}$$

with

$$\boldsymbol{A}_{n_{h,\mathcal{L}}+1,\mathcal{L}} = \boldsymbol{\Psi}_{n_{h,\mathcal{L}}+1,\mathcal{L}} \setminus \boldsymbol{\phi}_{\text{metrics},n_{h,\mathcal{L}},\mathcal{L}}, \tag{7.14}$$

$$\boldsymbol{B}_{\text{cond.},n_{h,\mathcal{L}},\mathcal{L}} = \boldsymbol{\Psi}_{\text{cond.},n_{h,\mathcal{L}},\mathcal{L}} \cup \boldsymbol{\phi}_{\text{metrics},n_{h,\mathcal{L}},\mathcal{L}}, \tag{7.15}$$

$$\begin{aligned} \boldsymbol{V}_{n_{h,\mathcal{L}}+1,\mathcal{L}} = \Big\{ &\psi \in Val\left(\boldsymbol{A}_{n_{h,\mathcal{L}}+1,\mathcal{L}}\right) : \\ &\varphi_{d_{n_{h,\mathcal{L}}+1}} > \varphi_{x_{n_{h,\mathcal{L}}}} - \varphi_{l_{n_{h,\mathcal{L}}}} - x_{\text{rear}} \Big\}. \end{aligned} \tag{7.16}$$

7.2.2 Selection of causality groups

The presented scenario generation model must now be modified to generate predominantly critical scenarios as required for IS. For that, the procedure described in section 5.2.1 is used. Therefore, the parametric family of distributions used for the optimization is defined as described in (5.19) and (5.21). The factor $p_{0,\text{irr.}}$ is of course still determined by (7.11). The factor $q_{0,\text{rel.}}$ however is proposed to be determined by

$$\begin{aligned} q_{0,\text{rel.}} &\left(\varphi_{\boldsymbol{\phi}_{\text{rel.}}} \middle| \varphi_{\mathfrak{v}} \right) = q_{0,\text{rel.}}^{(\text{beh.})} \left(\varphi_{\boldsymbol{\phi}_{\text{rel.}} \cap \boldsymbol{\phi}_{\text{beh.}}} \middle| \varphi_{\mathfrak{v}_{\text{beh.}}} \right) \\ &\cdot \prod_{\mathcal{L} \in \boldsymbol{L}} \prod_{i \in \boldsymbol{\nu}_{\varphi_\xi}^{(\text{rel.},\mathcal{L})}} q_{\text{metrics}}^{(\mathcal{L})} \left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\mathfrak{v}_{\text{init},\mathcal{L},i}} \right). \end{aligned} \tag{7.17}$$

$q_{0,\text{rel.}}^{(\text{beh.})}$ is a replicate of $q_{0,\text{rel.}}$ defined in section 5.3.2. To enable a good fitting of the distributions $q_{\text{metrics}}^{(\mathcal{L})}$, their form is chosen to be given by

$$q_{\text{metrics}}^{(\mathcal{L})} \left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} \middle| \varphi_{\mathfrak{v}_{\text{init},\mathcal{L},i}} \right) = \prod_{\mathfrak{m} \in \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} q_{0,\text{rel.},\mathfrak{m}} \left(\varphi_{\mathfrak{m}} \middle| \varphi_{\mathfrak{v}_{\text{init},\mathcal{L},i}^{(\mathfrak{m})}} \right). \tag{7.18}$$

The distributions $q_{0,\text{rel.},\mathfrak{m}}\left(\varphi_{\mathfrak{m}}\middle|\varphi_{\mathfrak{v}_{\text{init},\mathcal{L},i}^{(\mathfrak{m})}}\right)$ are truncated Cauchy mixture distributions as defined by (5.56). Their truncation intervals $\mathfrak{B}_{\mathfrak{m}}$ are equivalent to the intervals applied during the discretization in section 7.1.2. The CE optimization is performed by adopting (5.23) and (5.24) into algorithm 1 and by considering the modifications in section 5.2.2. As a result of (7.17) and (7.18), the causality groups according to (5.22) are

$$\mathfrak{C}_{\text{scenes}} = \boldsymbol{\phi}_{\text{rel.}} = (\boldsymbol{\phi}_{\text{rel.}} \cap \boldsymbol{\phi}_{\text{beh.}}) \cup \bigcup_{\mathcal{L}\in\boldsymbol{L}} \bigcup_{i\in\boldsymbol{\nu}_{\varphi_\xi}^{(\text{rel.},\mathcal{L})}} \boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}. \tag{7.19}$$

That is, all causality groups have been chosen to contain only one parameter.

Consequently, by distinguishing relevant and irrelevant parameters and by introducing the causality groups, one optimization of size $|\boldsymbol{\phi}| \cdot 9$ (assumption: 9 optimization parameters per simulation parameter) can be reduced to $|\mathfrak{C}_{\text{scenes}}| = |\boldsymbol{\phi}_{\text{rel.}}|$ optimizations of size 9 (Cauchy mixtures with 9 parameters per simulation parameter) . Here, the relation factor can be expressed by

$$\begin{aligned}\mathcal{K}_{\text{rel.}}\left(\varphi_h;\boldsymbol{\omega}\right) = &\prod_{\mathfrak{p}\in\boldsymbol{\phi}_{\text{rel.}}\cap\boldsymbol{\phi}_{\text{beh.}}} \frac{p_{\text{cauchy,trunc}}^{(\mathfrak{p})}\left(\varphi_{\mathfrak{p}}\middle|\mathbf{u}^{(\mathfrak{p})},\mathfrak{B}_{\mathfrak{p}}\right)}{q_{0,\text{rel.},\mathfrak{p}}\left(\varphi_{\mathfrak{p}}\middle|\boldsymbol{\omega}_{\text{beh.}}^{(\mathfrak{p})}\right)} \\ &\cdot \prod_{\mathcal{L}\in\boldsymbol{L}} \prod_{i\in\boldsymbol{\nu}_{\varphi_\xi}^{(\text{rel.},\mathcal{L})}} \frac{p_{\text{metrics}}^{(\mathcal{L})}\left(\varphi_{\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}}\middle|\varphi_{\boldsymbol{\Psi}_{\text{cond.},i,\mathcal{L}}}\right)}{\prod_{\mathfrak{m}\in\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}}} q_{0,\text{rel.},\mathfrak{m}}\left(\varphi_{\mathfrak{m}}\middle|\boldsymbol{\omega}_{\mathcal{L},i}^{(\mathfrak{m})}\right)}.\end{aligned} \tag{7.20}$$

Remember, that the values $\mathbf{u}^{(\mathfrak{p})}$ can be found in table 5.2 and that $p_{\text{metrics}}^{(\mathcal{L})}$ is an SPN which was directly learned from a dataset.

The causality group/relevant parameters used for the evaluation in the next section are listed in table 7.1.

7.2.3 Evaluation

The described optimization procedure is now illustrated. For that, the initial scenes are generated under the usage of $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}} = \{\Delta v_{i+1}, d_{i+1}\}$, which was the best metric set in the assessment in section 7.1.4. The networks are trained discriminatively only on d_{i+1}. The selection of this metric set can further be justified since it turned out that both initial velocity

Table 7.1 The relevant parameters $\boldsymbol{\phi}_{\text{rel.}}$ spanning the causality groups. For the definition of relative vehicles see fig. 5.2. For the concrete applied optimization, 7 parameters have been chosen empirically. The behavior related parameters are explained in table 5.2.

vehicle i	$\boldsymbol{\phi}_i \setminus \boldsymbol{\phi}_{\text{irr.}}$
$L_{\text{same},0}$	t_{react}, V_0, Δv_{i+1}, d_{i+1}
$L_{\text{same},1}$	V_0, Δv_{i+1}, d_{i+1}

differences and initial distances between consecutive vehicles are relevant for the criticality of a scenario. Hence, these properties must be controllable. Additionally, the use of a general safety metric such as RSS or TTC is not sufficient since these metrics only indirectly determine the relevant properties of the initial scenes. That means, it is for example possible that critical values of Δv_{i+1} (large) or critical values of d_{i+1} (small) occur for strongly uncritical values of RSS. This however would prevent a good optimization from taking place.

The rare event set whose occurrence rate shall be approximated is defined by

$$\boldsymbol{\varepsilon}_{\text{pre}} = \left\{ \varphi_\xi \in \boldsymbol{\Omega} : g_{\text{crit.}}^{(\text{RSS Dist,pre})} \left(\varphi_\xi, \boldsymbol{\mathcal{S}} \right) \leq \gamma \right\}, \tag{7.21}$$

whereas for $g_{\text{crit.}}^{(\text{RSS Dist,pre})}$, $\mathbf{A}_t^{(\varphi_\xi)} = \bigcup_{i \in \boldsymbol{\mathcal{S}}} \{(i, pre\,[i,t])\}$ and $\boldsymbol{\mathcal{S}} = \{ego\}$ holds deviating from the definitions in section 4.4.2. The used RSS parameters are given in table 5.6. The relevant parameters/ causality groups have been empirically chosen. They are listed in table 7.1.

Note that only predecessor vehicles in relation to the ego vehicle are considered relevant. The rationale is, that $\boldsymbol{\varepsilon}_{\text{pre}}$ only considers ego's collisions with its predecessor vehicle. The trailing vehicles thus should not be relevant and are therefore not optimized. As described in section 7.2.2, when considering a scenario with $n_{\text{veh}}^{(\varphi_\xi)} = 20$, this selection of relevant parameters allows reducing one optimization with $20 \cdot (11 + 2) \cdot 9 = 2340$ dimensions[6] to 7 optimizations of 9 dimensions.

The set of parameters used for CE optimization are listed in table 7.2. The simulation runs utilize time steps of $\Delta t = 0.04\,\text{s}$ and a time horizon

[6] It is $\boldsymbol{\phi} = 20 \cdot (11 + 2)$, because 11 behavior parameters and 2 scene parameters exist for each of the 20 vehicles.

Table 7.2 Parameters used for the optimization by algorithm 1. For $z_0, z_1, z_2, k_{\text{add}}$ and $\mathfrak{N}_{\text{add}}$, the values given in table 5.5 are reused.

$g_{\text{crit.}}$	$\mathfrak{N}$	η	$k_{\max}$	γ	β_1	β_2	
$g_{\text{crit.}}^{(\text{RSS Dist,pre})}$	10 000	0.02	100	-1	0.2	0.3	for $\boldsymbol{\phi}_{\text{beh.}} \cap \boldsymbol{\phi}_{\text{rel.}}$
					0.3	0.2	for $\boldsymbol{\phi}_{\text{scene}} \cap \boldsymbol{\phi}_{\text{rel.}}$

of $n_{\varphi_\xi} = 300$ steps. No relaxation phase is applied during the simulations. When executing the optimization, the η quantile converges to $\gamma_1 = -1$ after two iterations. Two additional optimization iterations (see section 5.2.2.3) are conducted afterwards to make sure a good approximation of the IS distribution has been found. The resulting parameter values are denoted by $\mathfrak{v}^*_{\text{RSS Dist,pre}}$.

The results of the optimization are visualized in figs. 7.8 and 7.9. Figures 7.8(c) and 7.9(b) ilustrate that the optimized distribution emphasizes small bin numbers of the metric d_{i+1} for $L_{\text{same},0}$ and $L_{\text{same},1}$. This is easily understood since small bin values correspond to small distances between consecutive vehicles and such small initial distances obviously render an initial scene more dangerous. Besides, another strong shift in the optimized distribution is observed for V_0 for vehicle $L_{\text{same},0}$, as displayed in fig. 7.8(a). Here, the distribution is shifted to low values for V_0. This makes sense since a low desired velocity V_0 increases $L_{\text{same},0}$'s desire to drive slowly and thereby to brake. Last but not least, a shift can be found in fig. 7.8(d). That is, the velocity difference now has an emphasis on larger values. This causes the sampling of more scenarios where $L_{\text{same},0}$ initially drives slower than ego. Obviously, this will also cause a higher collision risk. In contrast, t_{react} in fig. 7.8(b) is barely modified. The reason is, that this parameter makes it hard for $L_{\text{same},0}$ to react on actions of other vehicles. Therefore, this would mainly induce a collision risk between $L_{\text{same},0}$ and $L_{\text{same},1}$. However, the simulation[7] is only interested in collisions between ego and its surrounding vehicles. Consequently, no optimization of t_{react} on this uninteresting collision takes place. In total, it becomes clear, that $L_{\text{same},0}$ seems to have a stronger influence on criticality than $L_{\text{same},1}$ since the distribution shifts are either pronounced much stronger (for d_{i+1}) or there are no significant shifts for $L_{\text{same},1}$ at all (V_0 and Δv_{i+1}). This was expected and gives again

[7] to be more precise: the chosen metric $g_{\text{crit.}}^{(\text{RSS Dist,pre})}$

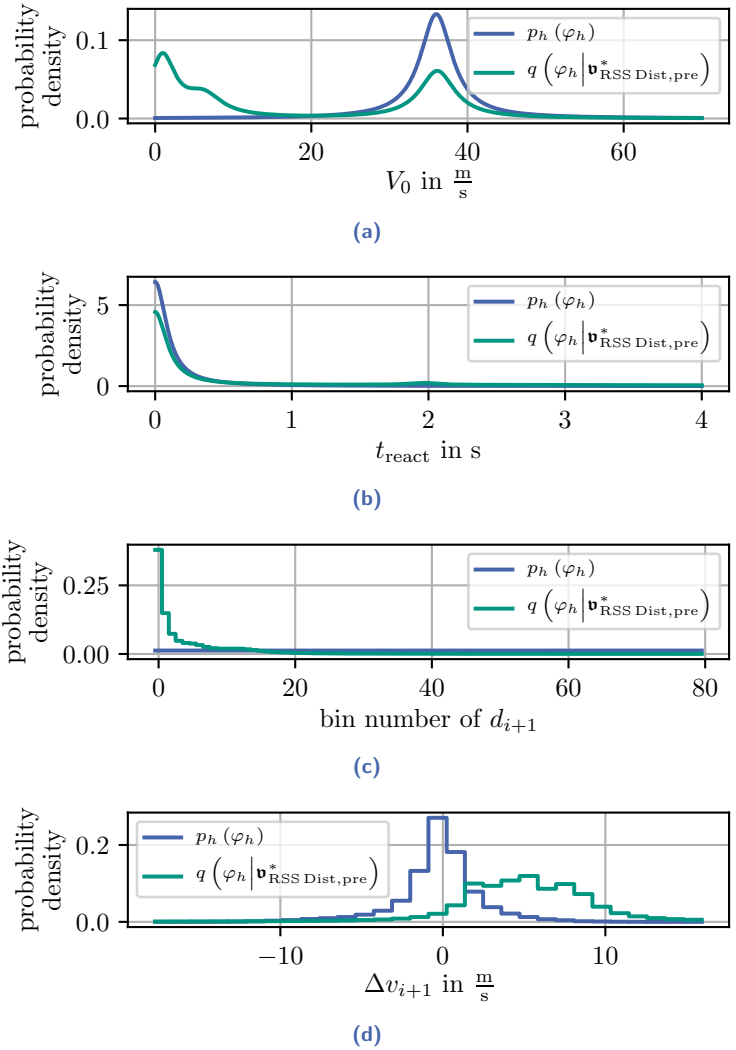


Figure 7.8 Marginals of p_h and q for the optimized parameters of $L_{\text{same},0}$ (see fig. 5.2). The distributions for the initial scene's velocity difference Δv_{i+1} and distance d_{i+1} (between the ego vehicle and $L_{\text{same},0}$) are given by histograms since their distribution $p^{(\mathcal{L})}_{\text{metrics}}$ is jointly determined by a discrete SPN in the unskewed case. The bins for d are chosen on the basis of quantiles and are not equidistantly distributed. Therefore, only the bin numbers and not real distance values are plotted on the x-axis of fig. 7.8(c). Small bin numbers correspond to small distance values. **(a)**, **(c)** and **(d)** were already published by Jesenski et al. [136]; © 2021 IEEE.

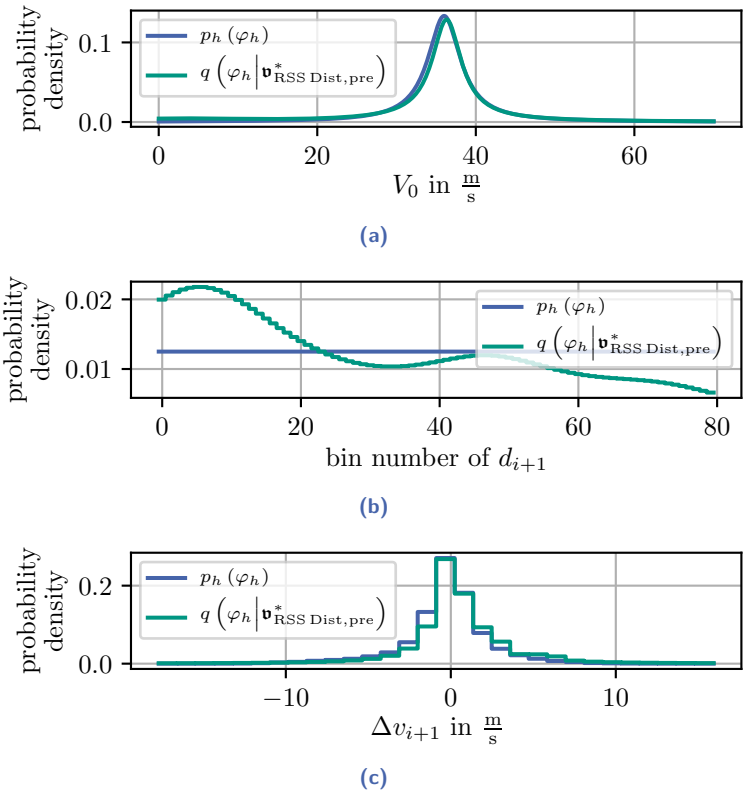


Figure 7.9 Marginals of $p_h\left(\varphi_h\right)$ and $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist,pre}}\right)$ for the optimized parameters of $L_{\text{same},1}$ (see fig. 5.2). Similar to fig. 7.8, Δv_{i+1} and distance d_{i+1} (both defined between $L_{\text{same},0}$ and $L_{\text{same},1}$) are given by histograms. (b) was already published by Jesenski et al. [136]; © 2021 IEEE.

confidence that the concept of relevant and irrelevant parts of a simulation run is a sensible approach.

After having accomplished and evaluated the optimization, 40 000 simulation runs are generated by the optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist,pre}}\right)$ and 330 000 scenarios by the original unskewed distribution $p_h\left(\varphi_h\right)$. Figure 7.10 shows a histogram over the criticalities of these generated simulation runs. As expected, the optimized distribution generates a much larger fraction of critical scenarios than the unskewed distribution.

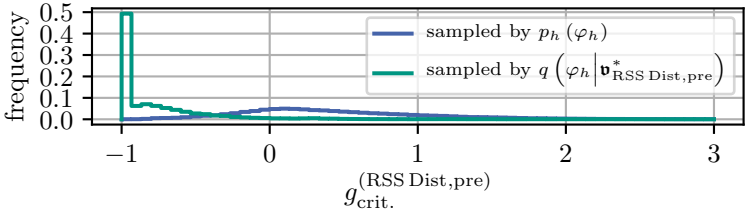


Figure 7.10 Histogram of criticality values of the simulation runs. The blue curve shows the dispensation of criticality values when sampling with the original unskewed distribution $p_h\left(\varphi_h\right)$. Obviously, most generated simulation runs are located at $g_{\text{crit.}}^{(\text{RSS Dist,pre})} \geq 0$. As discussed in section 4.4.2, this corresponds to totally safe conditions. However, when sampling by the optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist,pre}}\right)$ more critical scenarios are sampled. About 50 % of all runs even are located in the most critical bin next to $g_{\text{crit.}}^{(\text{RSS Dist,pre})} = -1$ which corresponds to a collision.

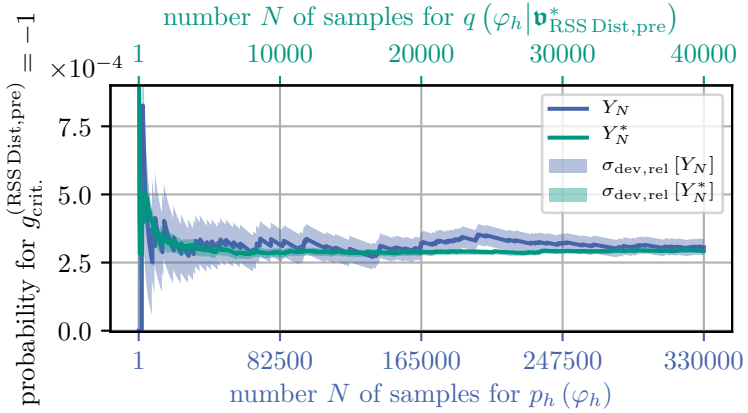


Figure 7.11 Collision rates approximated by sampling with the original distribution $p_h\left(\varphi_h\right)$ and by sampling with the optimized distribution $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist,pre}}\right)$. 330 000 samples were drawn by the original unskewed distribution and 40 000 samples were generated by using the optimized one. The image was already published by Jesenski et al. [136]; © 2021 IEEE.

Figure 7.11 proves that this higher amount of critical scenarios during the sampling by $q\left(\varphi_h \middle| \mathfrak{v}^*_{\text{RSS Dist,pre}}\right)$ allows extracting more relevant information with fewer simulation runs. It is shown, that the sampling by the optimized distribution gives a good approximation of the collision rate with significantly fewer samples. When comparing the standard deviations, the blue curve of the unskewed distribution has the same accuracy at sample 330 000 as the green curve at sample 2989. This corresponds to an acceleration factor of

$$\mathfrak{F}^{(\text{RSS Dist,pre})}_{\text{accel.}} \approx 110.4. \tag{7.22}$$

A collision rate of about

$$P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}_{\text{pre}}\right) \approx 2.92 \times 10^{-4}. \tag{7.23}$$

is obtained by the sampling from the optimized distribution.

7.3 Discussion/summary

In chapter 7, it was demonstrated how to modify the initial scene models from chapter 6 in order to sample mainly critical scenes. This works by conditioning on safety metrics. Among the tested safety metrics, the set $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}} = \{d_{i+1}, \Delta v_{i+1}\}$ was found to work best. Consequently, contribution 5 is satisfied.

In a further step, the developed critical scene model was combined with IS. This allows directly optimizing the initial scenes, which was not possible in chapter 5. It was shown, that this combination allows finding good approximations of the occurrence rate $P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}_{\text{pre}}\right)$. This worked without applying a relaxation phase which therefore becomes irrelevant. Hence, contribution 8 is accomplished.

Up to now, only longitudinal metrics ($g^{(\text{TTC})}_{\text{pair}}$, $g^{(\text{invTTC})}_{\text{pair}}$, $g^{(\text{RSS Dist})}_{\text{pair}}$, $\{d_{i+1}, \Delta v_{i+1}\}$) were incorporated into the initial scene models. In the future, the initial scene models and the respective metrics should be improved to consider lateral properties of a scene. Additionally, more advanced behavior models and ADF models should be tested. The entire framework should also be tested on more complicated roadway topologies like intersections. Chapter 6 already includes the groundwork for doing so. Besides, discriminative training methods (e.g. [5, 84]) might improve the accuracy of the trained version of $p^{(\mathcal{L})}_{\text{metrics}}$.

8 Conclusion & outlook

8.1 Conclusion

This work addresses the methodological challenge of validating highly automated driving functions of SAE levels 3 and higher by using simulations. Therefore, the work starts by giving a formalized overview about the validation tasks and simulation frameworks. Additionally, the current state of the art regarding simulation-based validation is discussed (for both, see chapter 3). The given overview is based on the 3-circles model [138].

As already shown in literature and recalled in section 2.3, one of the pressing issues for ADF validation is efficiency. This problem makes it impossible to validate ADFs by traditional methods like endurance runs alone (see section 2.3). The problem of efficiency translates to the application of simulations. When generating statistical evidence regarding an ADF's safety (e.g. collision rates), simulation runs are typically drawn statistically by distributions which represent reality. Since in reality only very few of the occurring scenarios will be relevant/critical for a good performing ADF, also a very high number of simulation runs is required in order to find enough relevant scenes to enable statistically stable simulation results.

It was shown in literature, that importance sampling (IS) can solve the efficiency problem (comp. section 5.1). However, ADF simulations can become quite complex which is challenging for IS since it scales very badly with the dimension of a simulation run. Up to now, IS therefore was limited to simulation runs with limited complexity or class.

As a first main part, this thesis therefore proposes a new procedure which enhances importance sampling's scalability by focusing the related CE optimization on the parts of a simulation run which are relevant with respect to the safety of the ADF-driven ego vehicle (see section 5.2). The method works by distinguishing between such relevant and irrelevant parts of a simulation run and additionally by categorizing the relevant parts into causality groups (procedure given in section 5.2.1). In general this method reduces the $|\mathfrak{v}|$-dimensional optimization, which is required for

finding the optimal importance sampling distribution, to $|\mathfrak{C}|$ optimizations with dimensions $\left|\mathfrak{v}^{(c)}\right|$. Remember that $\mathfrak{v}$ is the set of parameters required to parameterize the complete probability distribution which describes the simulation runs, $\mathfrak{C}$ is the set of causality groups and $\mathfrak{v}^{(c)} \subset \mathfrak{v}$ is the set of parameters which fully describes the part of the probability distribution belonging to causality group $c \in \mathfrak{C}$.

The proposed, scalable IS procedure was tested by applying it on a simulation framework which simulates complex scenarios on a 6-lane highway section (see section 5.3). In order to consider a realistic complexity of classes and scenarios, the initial scenes were directly drawn from a real-world dataset. The dynamics of the simulations were determined by simple parametric models (IDM and MOBIL). It could be shown that it is possible to accelerate the determination of the occurrence rate of critical events for an ADF-controlled vehicle in the middle lane of the highway scenario by factors of 107 and 496, respectively (see section 5.3.4). Note that the proposed method was able to reduce the dimensions of the required CE optimization from 2475 dimensions (which is not feasible) to 3 optimizations with 9 dimensions. Occurrence rates of $P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}_{\text{RSS Dist}}\right) \approx 5.98 \times 10^{-5}$ and $P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}_{\text{RSS Beh}}\right) \approx 2.79 \times 10^{-6}$ were obtained for the critical event sets $\boldsymbol{\varepsilon}_{\text{RSS Dist}}$ and $\boldsymbol{\varepsilon}_{\text{RSS Beh}}$. Since the initial scenes of the simulation framework were drawn from a pure dataset, it was impossible to optimize them within the CE approach, only the optimization of critical vehicle behavior could thus be achieved. This caused the necessity of a relaxation phase during this proof of concept of the method (see section 5.3.3) in order to decouple the initial scenes from the ADF's safety.

Additionally, the elementary effects method was shown to be efficient in identifying the relevant parameters in $\boldsymbol{\phi}$ which span the necessary causality groups $\mathfrak{C}$ (section 5.4). Such a method is necessary to make sure that on the one hand no relevant parameters are overlooked (which may lead to wrong results for the determined statistical evidence) and on the other hand as few as possible parameters are included during the optimizations (in order to keep the number of causality groups as well as their dimensions $\left|\mathfrak{v}^{(c)}\right|$ as small as possible).

As a second main part of the thesis, the current state of the art of initial scene models (see chapter 6), which are a prerequisite for enabling the optimization of the initial scenes and thereby for removing the unrealistic relaxation phase, is improved by this work. For that, previous research

by Wheeler et al. [119] which covered only straight highway sections (see section 6.1.1) was extended to complex topologies like intersections. This was illustrated by generating scenes for intersections included in the inD and the KoPER datasets (see section 6.2). This extension could be achieved by separately sampling so-called lane sections and by conditioning on related lane sections during the lane-section specific sampling.

Furthermore, the use of Bayesian networks as proposed by Wheeler et al. [119] scales badly with the network size. Consequently, BN-based sampling becomes slow or even infeasible for large scene models and/or complex topologies. As a solution, this work proposes using sum-product networks as an alternative to BNs (see section 6.3). Multiple SPN learning algorithms with different parameterizations were tested and compared to learning and sampling by a BN. In this context, it could be shown that the fastest SPN approach MSPN_{km} is about 36 times faster than a BN when sampling a complete highway section. On an Intel Xeon E-2144G with 64 GB of RAM, MSPN_{km} only needs about 2.44 s to sample a complete highway section, whereas the respective BN needs about 87.30 s. Furthermore, SPNs and BNs delivered a comparable modeling quality (see section 6.3.3).

As a third main part, the developed initial scene models were further adapted to enable their coupling with IS. That shall allow the optimization of the initial scenes when trying to find an optimal IS distribution, which was not yet possible in chapter 5. In order to make this coupling possible, the SPN-based scene generation approach was further modified in order to enable the sampling of predominantly critical scenes (see section 7.1). This could be achieved by conditioning on pair-wise safety metrics (for metrics see chapter 4). For that, the relevant distributions $p_{\text{cond.}}^{(\mathcal{L})}$ were trained discriminatively by MSPN_{km}. A comparison confirmed that this discriminative learning approach yields better results than pure generative learning (section 7.1.3). In addition, multiple pair-wise safety metrics were tested for creating the critical scenes. Eventually, the metric $\boldsymbol{\phi}_{\text{metrics},i,\mathcal{L}} = \{d_{i+1}, \Delta v_{i+1}\}$ was found to achieve the best results since it strongly shifts both Δv and d between consecutive vehicles in a lane (see section 7.1.4).

The resulting modified initial scene model was then incorporated into IS (see section 7.2). It could be experimentally shown (on a modified version of the simulation framework from section 5.3) that it is now possible to directly optimize on the generation of critical initial scenes. As a result, no relaxation phase was necessary anymore. Finally, a collision rate of

$P_{\text{crit}}\left(\xi \in \boldsymbol{\varepsilon}_{\text{pre}}\right) \approx 2.92 \times 10^{-4}$ could be determined in a strongly accelerated manner. Numerically, this application of IS accelerated the generation of the statistical result by a factor of about 110. The related CE optimization could be reduced from a dimension of 2340 to 7 optimizations with dimension 9.[1]

As a fourth main part, work on IS compatible metrics was executed. The respective requirements for the metrics are discussed in section 4.2. Firstly, it was shown that a physical RSS-based metric $g_{\text{crit.}}^{(\text{RSS Dist})}$ can be applied on IS (see sections 4.4.2 and 5.3.4.1). Secondly, a behavior-based safety metric $g_{\text{pair}}^{(\text{RSS Beh})}\left(p, t, \varphi_{\xi}\right)$ was developed (see section 4.4.3). This becomes necessary since traditional metrics only evaluate the physical criticality between two vehicles but not the fact which vehicle causes it. This is however a relevant question since an ADF cannot be blamed for an accident which it could not prevent. It was shown that $g_{\text{crit.}}^{(\text{RSS Beh})}$ fulfills IS's requirements for a safety metric. For that, it was applied on the proposed IS framework (see section 5.3.4.2).

The simulations have shown that a large difference in the occurrence probability between the respective sets $\boldsymbol{\varepsilon}_{\text{RSS Dist}}$, $\boldsymbol{\varepsilon}_{\text{RSS Beh}}$ and $\boldsymbol{\varepsilon}_{\text{pre}}$ exists. It is shown, that without IS, the tiny occurrence rate of $\boldsymbol{\varepsilon}_{\text{RSS Beh}}$ could not even have been determined by the simulations in feasible time (see section 5.3.4.2). Note that the occurrence rate of $\boldsymbol{\varepsilon}_{\text{pre}}$ is the highest, despite the fact that the underlying metric only considers the criticality to the predecessor of the ego vehicle. The reason for that is that the occurrence rates for $\boldsymbol{\varepsilon}_{\text{RSS Dist}}$ and $\boldsymbol{\varepsilon}_{\text{RSS Beh}}$ were evaluated in chapter 5 without optimizing the initial scenes. As already mentioned, that required the introduction of a relaxation phase which artificially reduces risk since it ditches a lot of dangerous initial scenes. In contrast, $\boldsymbol{\varepsilon}_{\text{pre}}$ was assessed in chapter 7 where a relaxation phase was not necessary anymore due to the coupling of an initial scene model.

In summary, this work proposes a new scalable and efficient, IS-based simulation framework for the validation of highly automated vehicles. Additionally, compatible initial scene models and a compatible behavior-based metric were investigated.

Finally, it must be noted that the developed framework is only applicable if it is possible to neglect some parameters in $\boldsymbol{\phi}$ in the optimization because they are not relevant for the assessed ADF's safety and/or if it is possible to categorize the relevant parameters into causality groups. If, in contrast, all of the parameters would have the same relevance for safety, the approach's

[1] Valid for an example scene with 20 vehicles. The scenes used by the framework and generated by the initial scene models of course contain a variable number of vehicles.

ability to reduce the dimension of the respective CE optimization decreases. However, since the influence of a surrounding vehicle on the ADF-driven ego vehicle decreases strongly when their separation increases or if another vehicle is located between them, the method should be applicable in reality (compare for example the decrease of relevance between $L_{\text{same},0}$ and $L_{\text{same},1}$ in section 7.2.3). Additionally, it is very important, that the used models' parameters allow optimizing/emphasizing all relevant parts of the scenario space. All used and potentially very complex models must be designed accordingly. If this is not possible, the occurrence rates of the assessed event set might be underestimated when only a limited number of simulation runs is drawn for determining the statistical evidence. This is of course the problem which also necessitated the introduction of the relaxation phase in chapter 5 which was circumvented by improving the models. This proves, that a thoughtful model design is absolutely essential.

8.2 Outlook

There are multiple open points left for analysis in future work.

1. The initial scene models should be improved by incorporating more information about lateral properties and angular alignment of the vehicles. This is of course also valid for the definition of the pair-wise metrics used to generate predominantly critical initial scenes.

2. More sophisticated behavior and ADF models should be tested within the framework. In an ideal case, the ADF-controlled ego vehicle is represented by as much production software code and hardware as possible.

3. IS should be tested on more complex road topologies. For that, the proposed, extended version of the initial scene models, which represent e.g. intersections, may be used.

4. With regard to the intersections and the high combinatorics of the resulting lane sections, even larger datasets would be beneficial for the training of the required SPNs/BNs for the extended initial scene models.

5. Up to now, only one-dimensional causality groups (each causality group only included one parameter) were examined. These one-

dimensional causality groups might not be sufficient when simulating more complex scenarios and behavior models. Additionally, it would in general be interesting to find out if more complicated causality groups can further improve the efficiency and accuracy of the developed, scalable IS approach. Hence, new types of causality groups should be created and tested. In this relation, it would also be beneficial to have a further look at sensitivity analysis in order to assess the interactions between the relevant parameters. This assessment might then be used to create causality groups of maximally interacting parameters (Remember, that parameters in different causality groups are assumed to be approximately independent of each other with regard to the optimal importance sampling distribution).

Appendix

A Derivation of confidence level for test driving

For the sake of completeness, this appendix shall present a derivation for formula (2.3). Bayes' rule implies

$$p(\varphi_r|\varphi_n,\varphi_\mathrm{i}) = p(\varphi_\mathrm{i}|\varphi_n,\varphi_r)\cdot\frac{p(\varphi_r|\varphi_n)}{p(\varphi_\mathrm{i}|\varphi_n)}. \tag{A.1}$$

Additionally, it is possible to marginalize

$$p(\varphi_\mathrm{i}|\varphi_n) = \int_0^1 p(\varphi_\mathrm{i}|\varphi_n,\varphi_r)\cdot p(\varphi_r|\varphi_n)\,\mathrm{d}\varphi_r. \tag{A.2}$$

Combining (A.1), (A.2) and using the a-priori distribution $p(\varphi_r|\varphi_n) = const.$ (constant uniform distribution) results in

$$\begin{aligned} p(\varphi_r|\varphi_n,\varphi_\mathrm{i}) &= \frac{p(\varphi_\mathrm{i}|\varphi_n,\varphi_r)}{\int_0^1 p(\varphi_\mathrm{i}|\varphi_n,\varphi_r)\,\mathrm{d}\varphi_r} \\ &= \frac{p(\varphi_\mathrm{i}|\varphi_n,\varphi_r)}{\binom{\varphi_n}{\varphi_\mathrm{i}}\cdot\mathcal{B}(x=\varphi_n-\varphi_\mathrm{i}+1, y=\varphi_\mathrm{i}+1)}. \end{aligned} \tag{A.3}$$

Here, $\mathcal{B}(x,y)$ is Euler's beta function. By using the identity ([17], p. 1114)

$$\mathcal{B}(x,y) = \frac{(x-1)!(y-1)!}{(x+y-1)!}, \tag{A.4}$$

it can be shown that

$$p(\varphi_r|\varphi_n,\varphi_\mathrm{i}) = (\varphi_n+1)\cdot p(\varphi_\mathrm{i}|\varphi_n,\varphi_r). \tag{A.5}$$

By applying (2.1) and (A.5),

$$C(R|\varphi_n,\varphi_\mathrm{i}) = \int_R^1 p(\varphi_r|\varphi_n,\varphi_\mathrm{i})\,\mathrm{d}\varphi_r$$

$$= 1 - \binom{\varphi_n+1}{\varphi_\mathrm{i}} \cdot (\varphi_n+1-\varphi_\mathrm{i}) \cdot \int_0^R \varphi_r^{\varphi_n-\varphi_\mathrm{i}}(1-\varphi_r)^{\varphi_\mathrm{i}}\,\mathrm{d}\varphi_r \quad \text{(A.6)}$$

is obtained. With the identity

$$\sum_{j=0}^{|w|}\binom{s}{j}p^j(1-p)^{s-j} = (s-w)\binom{s}{w}\cdot\int_0^{1-p} t^{s-w-1}\cdot(1-t)^w\mathrm{d}t, \quad \text{(A.7)}$$

(A.6) changes to

$$C(R|\varphi_n,\varphi_\mathrm{i}) = 1 - \sum_{j=0}^{|\varphi_\mathrm{i}|}\binom{\varphi_n+1}{j}(1-R)^j R^{\varphi_n+1-j}. \quad \text{(A.8)}$$

When, in the best case, $\varphi_\mathrm{i} = 0$ failures happen during the test run, this can be simplified to

$$C(R|\varphi_n,0) = 1 - R^{\varphi_n+1}. \quad \text{(A.9)}$$

Therefore, the number of necessary kilometers can be determined by

$$\varphi_n = \frac{\ln(1-C(R|\varphi_n,0))}{\ln(R)} - 1 \approx \frac{\ln(1-C(R|\varphi_n,0))}{\ln(R)}. \quad \text{(A.10)}$$

B Datasets

B.1 Ko-PER dataset

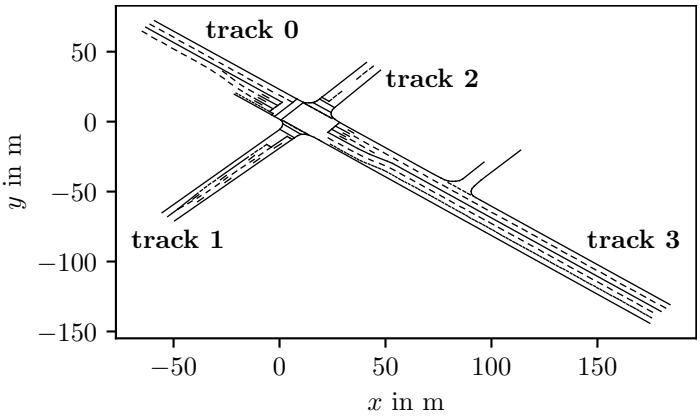
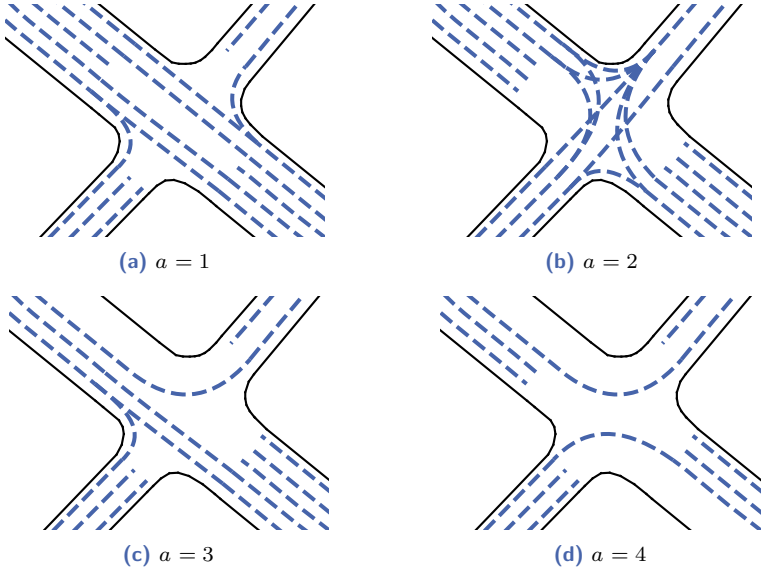


Figure B.1 Digital map of the KoPER intersection in Aschaffenburg as provided in the dataset.

Strigel et al. [99] introduced the KoPER dataset in 2014. The dataset is accessible for free and provides 6 min 28 s of manually labeled object-data in an intersection which is located in the city of Aschaffenburg, Germany. The intersection was observed by laserscanners and video cameras which are installed at infrastructure components. The sensors detect incoming traffic, but have a very limited field of view on vehicles leaving the intersection. The data provide a time resolution of 12.5 Hz and include the position, width, length, orientation angle and classification of the vehicles and pedestrians moving through the intersection. A digital model of the intersection is also provided (see fig. B.1). There are four driveways into the intersection which shall be denoted as track 0 to track 3. The majority of vehicles is driving from track 0 to track 3 and vice versa. The traffic flow is controlled by



(a) $a = 1$ (b) $a = 2$

(c) $a = 3$ (d) $a = 4$

Figure B.2 The blue dashed lines in the images show where it is allowed to drive simultaneously. Sub-figures (a), (b), (c) and (d) give the lanes for the different traffic light phases. The images are based on figure 11 by Jesenski et al. [132]; © 2019 IEEE.

traffic lights which can switch between four different traffic light phases a. The effects of the traffic light phases are illustrated in fig. B.2. The traffic rules allow the vehicles to drive on the blue dashed lane sections. Note that pedestrians are filtered from the KoPER data before using the dataset in this work.

B.2 highD dataset

The highD dataset contains the movement of road users at straight sections of the German Autobahn. Krajewski et al. [61] published the dataset in 2018. The data were taken at six different locations on highways around Cologne in the years 2017 and 2018. The recordings have a total length of 16.5 h and were taken by a camera mounted to a drone which was flying over the respective highway section. Therefore, the data are free

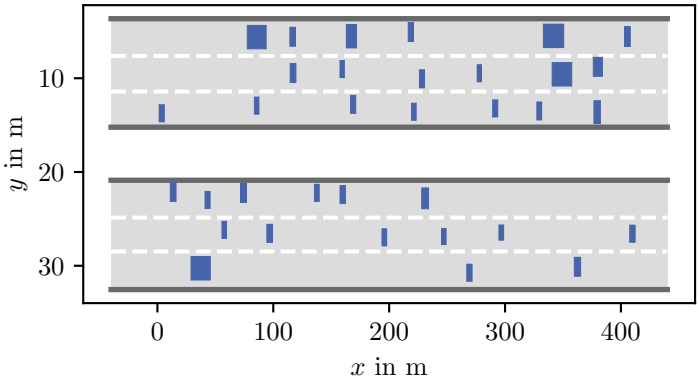


Figure B.3 A typical scene taken from the highD dataset at locationId 1. The rectangles represent bounding boxes of the vehicle objects which were preprocessed by computer vision and tracking algorithms. Image originally published by Jesenski et al. [135]; © 2020 IEEE.

of occlusions caused by traffic participants. The length of the recorded highway sections is between 400 to 420 m. The data were taken at a 4k resolution at 25 Hz. The recorded raw data were already processed by utilizing tracking and computer vision algorithms. Thus, processed object lists annotated with the properties of the driven tracks are ready for usage. A typical time snapshot (scene) in the dataset is illustrated in fig. B.3. As already mentioned, the data were taken at different highway locations with different road topologies. Since for the purpose of this thesis data with the same topology are required, the location with the longest measurement time was selected and the rest of the data were neglected. Consequently highD's locationId 1, which made up about 11 h of the data, was used in this work. The global coordinates given in the dataset were transformed into lane related Frenet coordinates (compare section 6.2.2.1). Afterwards, the data were discretized according to section 6.3.1.

B.3 inD dataset

In 2019, Bock et al. [14] published the inD dataset. It contains trajectories of vehicles at four intersections in the city of Aachen, Germany. The recorded

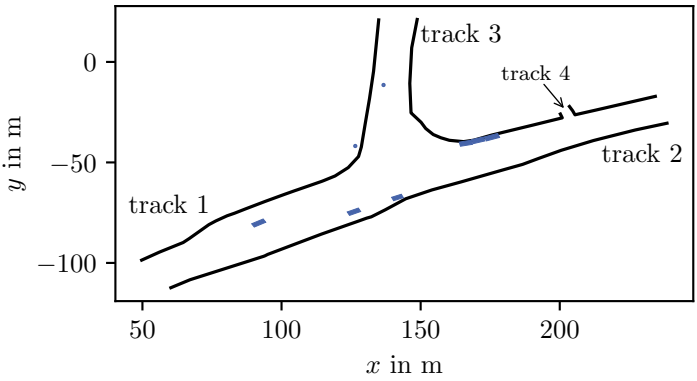


Figure B.4 A typical scene taken from inD records of Neuköllner Straße, Aachen. The rectangles represent bounding boxes of vehicles. The blue dots show the locations of pedestrians.

data have a length of about 10 h. The data were recorded by a camera mounted to a drone which was positioned at altitudes up to 100 m over the respective intersection. The used camera has a 4k resolution and takes images at 25 Hz. The recorded intersections have a size between 80 m × 40 m and 140 m × 70 m. Note that the intersections are unsignalized and have a maximum allowed speed of 50 $\frac{\text{km}}{\text{h}}$. Pedestrians, cyclists, cars, trucks, as well as buses were recorded. The data were already pre-processed by the use of computer vision and tracking algorithms. Therefore, the dataset delivers simple-to-use and already classified objects with referenced tracks. In the context of this work, only the data recorded at Neuköllner Straße, Aachen were used. The reason is that this work focuses on vehicles and neglects pedestrians. Neuköllner Straße contains comparatively few VRUs in comparison to the total amount of included traffic participants. At Neuköllner Straße, a double-laned priority road intersects with a single-laned side road as shown in the exemplary scene in fig. B.4. The four entrances into the intersection are label by *track 1* to *track 4*. 1.83 h of data were recorded at Neuköllner Straße. Only objects classified as car, bus or truck were used within this work. The rest of the objects were filtered out.

Bibliography

[1] **Abbas, Houssam, O'Kelly, Matthew E., Rodionova, Alena, and Mangharam, Rahul**. *A Driver's License Test for Driverless Vehicles*. In: *Mechanical Engineering* 139.12 (1, 2017), pp. 13–16.

[2] **Abdessalem, Raja Ben, Nejati, Shiva, Briand, Lionel C., and Stifter, Thomas**. *Testing advanced driver assistance systems using multi-objective search and neural networks*. In: 2016 31st IEEE/ACM International Conference on Automated Software Engineering (ASE). Singapore, 2016.

[3] **Abdessalem, Raja Ben, Panichella, Annibale, Nejati, Shiva, Briand, Lionel C., and Stifter, Thomas**. *Testing autonomous cars for feature interaction failures using many-objective search*. In: 2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE). Montpellier, France: ACM Press, 2018, pp. 143–154.

[4] **Abdessalem, Raja Ben, Nejati, Shiva, Briand, Lionel C., and Stifter, Thomas**. *Testing vision-based control systems using learnable evolutionary algorithms*. In: 2018 IEEE/ACM 40th International Conference on Software Engineering (ICSE). Gothenburg, Sweden: ACM, 27, 2018, pp. 1016–1026.

[5] **Adel, Tameem, Balduzzi, David, and Ghodsi, Ali**. *Learning the Structure of Sum-Product Networks via an SVD-Based Algorithm*. In: *Proceedings of the Thirty-First Conference on Uncertainty in Artificial Intelligence*. UAI'15. Amsterdam, Netherlands: AUAI Press, 2015, pp. 32–41.

[6] **Allen, Brian L, Shin, B Tom, and Cooper, Peter J**. *Analysis of Traffic Conflicts and Collisions*. In: *Transportation Research Record* 667 (1978), pp. 67–74.

[7] **Althoff, Matthias and Lutz, Sebastian**. *Automatic Generation of Safety-Critical Test Scenarios for Collision Avoidance of Road Vehicles*. In: 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, China: IEEE, 2018, pp. 1326–1333.

[8] **Ankan, Ankur and Panda, Abinash**. *pgmpy: Probabilistic graphical models using python*. In: *Proceedings of the 14th Python in Science Conference (SCIPY 2015)*. 2015.

[9] **Arechiga, Nikos**. *Specifying Safety of Autonomous Vehicles in Signal Temporal Logic*. In: 2019 IEEE Intelligent Vehicles Symposium (IV). Paris, France: IEEE, 2019, pp. 58–63.

[10] **Bagschik, Gerrit, Menzel, Till, and Maurer, Markus**. *Ontology based Scene Creation for the Development of Automated Vehicles*. In: 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, China: IEEE, 2018, pp. 1813–1820.

[11] **Batsch, Felix, Daneshkhah, Alireza, Cheah, Madeline, Kanarachos, Stratis, and Baxendale, Anthony**. *Performance Boundary Identification for the Evaluation of Automated Vehicles using Gaussian Process Classification*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE, 2019, pp. 419–424.

[12] **Beck, James L. and Zuev, Konstantin M.** *Rare-Event Simulation*. In: *Handbook of Uncertainty Quantification*. Ed. by **Ghanem, Roger, Higdon, David, and Owhadi, Houman**. DOI: 10.1007/978-3-319-11259-6_24-1. Cham: Springer International Publishing, 2015, pp. 1–26.

[13] **Beglerovic, Halil, Stolz, Michael, and Horn, Martin**. *Testing of autonomous vehicles using surrogate models and stochastic optimization*. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). Yokohama, Japan: IEEE, 2017, pp. 1–6.

[14] **Bock, Julian, Krajewski, Robert, Moers, Tobias, Runde, Steffen, Vater, Lennart, and Eckstein, Lutz**. *The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections*. In: *arXiv preprint arXiv:1911.07602*. 2019.

[15] **Boer, Pieter-Tjerk de, Kroese, Dirk P., Mannor, Shie, and Rubinstein, Reuven Y.** *A Tutorial on the Cross-Entropy Method*. In: *Annals of Operations Research* 134.1 (2005), pp. 19–67.

[16] **Bonsall, Peter, Liu, Ronghui, and Young, William**. *Modelling safety-related driving behaviour—impact of parameter values*. In: *Transportation Research Part A: Policy and Practice* 39.5 (2005), pp. 425–444.

[17] **Bronštejn, Ilja N., Semendjaev, Konstantin A., Musiol, Gerhard, and Mühlig, Heiner**, eds. *Taschenbuch der Mathematik*. 8., vollst. überarb. Aufl. OCLC: 808683094. Frankfurt am Main: Deutsch, 2012. 1230 pp.

[18] **Bühler, Oliver and Wegener, Joachim**. *Automatic Testing of an Autonomous Parking System Using Evolutionary Computation*. In: SAE 2004 World Congress & Exhibition. SAE International, 8, 2004.

[19] **Campolongo, Francesca, Cariboni, Jessica, and Saltelli, Andrea**. *An effective screening design for sensitivity analysis of large models*. In: *Environmental Modelling & Software* 22.10 (2007), pp. 1509–1518.

[20] **Chen, Yi-Chun, Wheeler, Tim A., and Kochenderfer, Mykel J.** *Learning Discrete Bayesian Networks from Continuous Data*. In: *Journal of Artificial Intelligence Research* 59 (22, 2017), pp. 103–132.

[21] **Corso, Anthony and Kochenderfer, Mykel J.** *Interpretable Safety Validation for Autonomous Vehicles*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). Rhodes, Greece (Virtual): IEEE, 2020, pp. 2448–2453.

[22] **Corso, Anthony, Lee, Ritchie, and Kochenderfer, Mykel J.** *Scalable Autonomous Vehicle Safety Validation through Dynamic Programming and Scene Decomposition*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). Rhodes, Greece (Virtual): IEEE, 2020, pp. 2442–2447.

[23] **Corso, Anthony, Du, Peter, Driggs-Campbell, Katherine, and Kochenderfer, Mykel J.** *Adaptive Stress Testing with Reward Augmentation for Autonomous Vehicle Validation*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE, 2019, pp. 163–168.

[24] **Dennis, Aaron and Ventura, Dan**. *Greedy structure search for sum-product networks*. In: *Proceedings of the 24th International Conference on Artificial Intelligence*. IJCAI'15. Buenos Aires, Argentina: AAAI Press, 2015, pp. 932–938.

[25] **Dennis, Aaron and Ventura, Dan**. *Online Structure-Search for Sum-Product Networks*. In: 2017 16th IEEE International Conference on Machine Learning and Applications (ICMLA). Cancun, Mexico: IEEE, 2017, pp. 155–160.

[26] **Eggert, Julian**. *Predictive risk estimation for intelligent ADAS functions*. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). Qingdao, China: IEEE, 2014, pp. 711–718.

[27] **Elrofai, Hala, Worm, Daniël, and Op den Camp, Olaf**. *Scenario Identification for Validation of Automated Driving Functions*. In: *Advanced Microsystems for Automotive Applications 2016*. Ed. by **Schulze, Tim, Müller, Beate, and Meyer, Gereon**. DOI: 10.1007/978-3-319-44766-7_13. Cham: Springer International Publishing, 2016, pp. 153–163.

[28] **Erdmann, Jakob**. *Lane-Changing Model in SUMO*. In: *Proceedings of the SUMO2014 Modeling Mobility with Open Data*. Vol. 24. Reports of the DLR-Institute of Transportation SystemsProceedings. Berlin, Germany, 2014, pp. 77–88.

[29] **Fainekos, Georgios E. and Pappas, George J.** *Robustness of temporal logic specifications for continuous-time signals*. In: *Theoretical Computer Science* 410.42 (2009), pp. 4262–4291.

[30] **Gelder, Erwin de and Paardekooper, Jan-Pieter**. *Assessment of Automated Driving Systems using real-life scenarios*. In: 2017 IEEE Intelligent Vehicles Symposium (IV). Los Angeles, CA, USA: IEEE, 2017, pp. 589–594.

[31] **Gelder, Erwin de, Paardekooper, Jan-Pieter, Op den Camp, Olaf, and De Schutter, Bart**. *Safety assessment of automated vehicles: how to determine whether we have collected enough field data?* In: *Traffic Injury Prevention* 20 (sup1 12, 2019), pp. 162–170.

[32] **Gens, Robert and Domingos, Pedro**. *Learning the Structure of Sum-Product Networks*. In: *Proceedings of the 30th International Conference on Machine Learning*. Atlanta, Georgia, USA: PMLR, 2013, pp. 873–880.

[33] **Gietelink, O., De Schutter, B., and Verhaegen, M.** *Adaptive importance sampling for probabilistic validation of advanced driver assistance systems*. In: 2006 American Control Conference. Minneapolis, MN, USA: IEEE, 2006, pp. 4002–4007.

[34] **Gietelink, O.J., Ploeg, J., De Schutter, B., and Verhaegen, M.** *Development of a driver information and warning system with vehicle hardware-in-the-loop simulations*. In: *Mechatronics* 19.7 (2009), pp. 1091–1104.

[35] **Gindele, Tobias, Brechtel, Sebastian, and Dillmann, Rudiger**. *Learning Driver Behavior Models from Traffic Observations for Decision Making and Planning*. In: *IEEE Intelligent Transportation Systems Magazine* 7.1 (2015), pp. 69–79.

[36] **Gipps, P.G.** *A behavioural car-following model for computer simulation*. In: *Transportation Research Part B: Methodological* 15.2 (1981), pp. 105–111.

[37] **Grindal, Mats, Offutt, Jeff, and Andler, Sten F.** *Combination testing strategies: a survey*. In: *Software Testing, Verification and Reliability* 15.3 (2005), pp. 167–199.

[38] **Guyon, Isabelle and Elisseeff, André**. *An introduction to variable and feature selection*. In: *Journal of machine learning research* 3 (2003), pp. 1157–1182.

[39] **Hallerbach, Sven, Xia, Yiqun, Eberle, Ulrich, and Koester, Frank**. *Simulation-Based Identification of Critical Scenarios for Cooperative and Automated Vehicles*. In: *SAE International Journal of Connected and Automated Vehicles* 1.2 (3, 2018), pp. 93–106.

[40] **Hayward, John C**. *Near-miss determination through use of a scale of danger*. In: 51st Annual Meeting of the Highway Research Board. Washington District of Columbia, United States: Highway Research Board, 1972, pp. 24–34.

[41] **Hekmatnejad, Mohammad, Hoxha, Bardh, and Fainekos, Georgios**. *Search-based Test-Case Generation by Monitoring Responsibility Safety Rules*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). Rhodes, Greece (Virtual): IEEE, 2020, pp. 2454–2461.

[42] **Helmer, Thomas**. *Development of a Methodology for the Evaluation of Active Safety using the Example of Preventive Pedestrian Protection*. Springer Theses. DOI: 10.1007/978-3-319-12889-4. Cham: Springer International Publishing, 2015.

[43] **Hempen, Thomas, Biank, Sanjana, Huber, Werner, and Diedrich, Christian**. *Model Based Generation of Driving Scenarios*. In: *Intelligent Transport Systems – From Research and Development to the Market Uptake*. Ed. by **Kováčiková, Tatiana, Buzna, Ľuboš, Pourhashem, Ghadir, Lugano, Giuseppe, Cornet, Yannick, and Lugano, Nathalie**. Vol. 222. DOI: 10.1007/978-3-319-93710-6_17. Cham: Springer International Publishing, 2018, pp. 153–163.

[44] **Hillenbrand, J., Kroschel, K., and Schmid, V.** *Situation assessment algorithm for a collision prevention assistant*. In: IEEE Proceedings. Intelligent Vehicles Symposium, 2005. Las Vegas, NV, USA: IEEE, 2005, pp. 459–465.

[45] **Huang, Zhiyuan, Lam, Henry, and Zhao, Ding**. *Towards affordable on-track testing for autonomous vehicle — A Kriging-based statistical approach*. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). Yokohama, Japan: IEEE, 2017, pp. 1–6.

[46] **Huang, Zhiyuan, Arief, Mansur, Lam, Henry, and Zhao, Ding**. *Evaluation Uncertainty in Data-Driven Self-Driving Testing*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE, 2019, pp. 1902–1907.

[47] *J3016: Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles*. SAE International, 2021.

[48] **Jenkins, Ian Rhys, Gee, Ludvig Oliver, Knauss, Alessia, Yin, Hang, and Schroeder, Jan**. *Accident Scenario Generation with Recurrent Neural Networks*. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). Maui, HI, USA: IEEE, 2018, pp. 3340–3345.

[49] **Julian, Kyle D., Lee, Ritchie, and Kochenderfer, Mykel J.** *Validation of Image-Based Neural Network Controllers through Adaptive Stress Testing*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). Rhodes, Greece (Virtual): IEEE, 2020, pp. 1007–1013.

[50] **Junietz, Philipp, Steininger, Udo, and Winner, Hermann**. *Macroscopic Safety Requirements for Highly Automated Driving*. In: *Transportation Research Record: Journal of the Transportation Research Board* 2673.3 (2019), pp. 1–10.

[51] **Kalra, Nidhi and Paddock, Susan**. *Driving to Safety: How Many Miles of Driving Would It Take to Demonstrate Autonomous Vehicle Reliability?* DOI: 10.7249/RR1478. RAND Corporation, 2016.

[52] **Kesting, Arne, Treiber, Martin, and Helbing, Dirk**. *General Lane-Changing Model MOBIL for Car-Following Models*. In: *Transportation Research Record: Journal of the Transportation Research Board* 1999.1 (2007), pp. 86–94.

[53] **Klischat, Moritz and Althoff, Matthias**. *Generating Critical Test Scenarios for Automated Vehicles with Evolutionary Algorithms*. In: 2019 IEEE Intelligent Vehicles Symposium (IV). Paris, France: IEEE, 2019, pp. 2352–2358.

[54] **Klueck, Florian, Li, Yihao, Nica, Mihai, Tao, Jianbo, and Wotawa, Franz**. *Using Ontologies for Test Suites Generation for Automated and Autonomous Driving Functions*. In: 2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW). Memphis, TN, USA: IEEE, 2018, pp. 118–123.

[55] **Koenig, Alexander, Witzlsperger, Kathrin, Leutwiler, Florin, and Hohmann, Sören**. *Overview of HAD validation and passive HAD as a concept for validating highly automated cars*. In: *at - Automatisierungstechnik* 66.2 (23, 2018), pp. 132–145.

[56] **Koller, Daphne and Friedman, Nir**. *Probabilistic graphical models: principles and techniques*. Adaptive computation and machine learning. Cambridge, MA: MIT Press, 2009. 1231 pp.

[57] **Koren, Mark and Kochenderfer, Mykel J.** *Efficient Autonomy Validation in Simulation with Adaptive Stress Testing*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE, 2019, pp. 4178–4183.

[58] **Koren, Mark and Kochenderfer, Mykel J.** *Adaptive Stress Testing without Domain Heuristics using Go-Explore*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). Rhodes, Greece (Virtual), 2020, pp. 1525–1530.

[59] **Koren, Mark, Alsaif, Saud, Lee, Ritchie, and Kochenderfer, Mykel J.** *Adaptive Stress Testing for Autonomous Vehicles*. In: 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, China: IEEE, 2018, pp. 1–7.

[60] **Koschi, Markus, Pek, Christian, Maierhofer, Sebastian, and Althoff, Matthias**. *Computationally Efficient Safety Falsification of Adaptive Cruise Control Systems*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE, 2019, pp. 2879–2886.

[61] **Krajewski, Robert, Bock, Julian, Kloeker, Laurent, and Eckstein, Lutz**. *The highD Dataset: A Drone Dataset of Naturalistic Vehicle Trajectories on German Highways for Validation of Highly Automated Driving Systems*. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). Maui, HI, USA: IEEE, 2018, pp. 2118–2125.

[62] **Krauß, Stefan**. *Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics*. doctoralthesis. Köln: Universität zu Köln, 1998. 115 pp.

[63] **Kuefler, Alex and Kochenderfer, Mykel J.** *Burn-In Demonstrations for Multi-Modal Imitation Learning*. In: 17th International Conference on Autonomous Agents and MultiAgentSystems. International Foundation for Autonomous Agents and Multiagent Systems, 2018, pp. 1071–1078.

[64] **Kuefler, Alex, Morton, Jeremy, Wheeler, Tim, and Kochenderfer, Mykel**. *Imitating driver behavior with generative adversarial networks*. In: 2017 IEEE Intelligent Vehicles Symposium (IV). Los Angeles, CA, USA: IEEE, 2017, pp. 204–211.

[65] **Lefevre, Stephanie, Sun, Chao, Bajcsy, Ruzena, and Laugier, Christian**. *Comparison of parametric and non-parametric approaches for vehicle speed prediction*. In: 2014 American Control Conference. Portland, OR, USA: IEEE, 2014, pp. 3494–3499.

[66] **Li, Sisi, Wang, Wenshuo, Mo, Zhaobin, and Zhao, Ding**. *Cluster Naturalistic Driving Encounters Using Deep Unsupervised Learning*. In: 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, China: IEEE, 2018, pp. 1354–1359.

[67] **Lopez, Pablo Alvarez, Wiessner, Evamarie, Behrisch, Michael, Bieker-Walz, Laura, Erdmann, Jakob, Flotterod, Yun-Pang, Hilbrich, Robert, Lucken, Leonhard, Rummel, Johannes, and Wagner, Peter**. *Microscopic Traffic Simulation using SUMO*. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). Maui, HI, USA: IEEE, 2018, pp. 2575–2582.

[68] **Lopez-Paz, David, Hennig, Philipp, and Schöllkopf, Bernhard**. *The Randomized Dependence Coefficient*. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems*. Vol. 1. NIPS'13. Lake Tahoe, Nevada, USA, 2013.

[69] **Mahmud, S.M. Sohel, Ferreira, Luis, Hoque, Md. Shamsul, and Tavassoli, Ahmad**. *Application of proximal surrogate indicators for safety evaluation: A review of recent developments and research needs*. In: *IATSS Research* 41.4 (2017), pp. 153–163.

[70] **Mitschke, Manfred and Wallentowitz, Henning**. *Dynamik der Kraftfahrzeuge*. DOI: 10.1007/978-3-658-05068-9. Wiesbaden: Springer Fachmedien Wiesbaden, 2014.

[71] **Molina, Alejandro, Vergari, Antonio, Di Mauro, Nicola, Natarajan, Sriraam, Esposito, Floriana, and Kersting, Kristian**. *Mixed Sum-Product Networks: A Deep Architecture for Hybrid Domains*. In: AAAI Conference on Artificial Intelligence. 2018.

[72] **Molina, Alejandro, Vergari, Antonio, Stelzner, Karl, Peharz, Robert, Subramani, Pranav, Di Mauro, Nicola, Poupart, Pascal, and Kersting, Kristian**. *SPFlow: An easy and extensible library for deep probabilistic learning using sum-product networks*. In: eprint arXiv:1901.03704. 2019.

[73] **Morris, Max D.** *Factorial Sampling Plans for Preliminary Computational Experiments*. In: *Technometrics* 33.2 (1991), pp. 161–174.

[74] **Morton, Jeremy, Wheeler, Tim A., and Kochenderfer, Mykel J.** *Analysis of Recurrent Neural Networks for Probabilistic Modeling of Driver Behavior*. In: *IEEE Transactions on Intelligent Transportation Systems* 18.5 (2017), pp. 1289–1298.

[75] **Mullins, Galen E., Stankiewicz, Paul G., and Gupta, Satyandra K.** *Automated generation of diverse and challenging scenarios for test and evaluation of autonomous vehicles*. In: 2017 IEEE International Conference on Robotics and Automation (ICRA). Singapore, Singapore: IEEE, 2017, pp. 1443–1450.

[76] **Neapolitan, Richard E.** *Learning Bayesian networks*. Prentice Hall series in artificial intelligence. OCLC: ocm52534097. Upper Saddle River, NJ: Pearson Prentice Hall, 2004. 674 pp.

[77] **Nitsche, P., Welsh, R.H., Genser, A., and Thomas, P.D.** *A novel, modular validation framework for collision avoidance of automated vehicles at road junctions*. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). Maui, HI, USA: IEEE, 2018, pp. 90–97.

[78] **Norden, Justin, O'Kelly, Matthew, and Sinha, Aman**. *Efficient Black-box Assessment of Autonomous Vehicle Safety*. In: arXiv. 2019.

[79] **O'Connor, Patrick D. T. and Kleyner, Andre**. *Practical reliability engineering*. 5th ed. OCLC: 775100516. Chicester: Wiley, 2012. 484 pp.

[80] **O'Kelly, Matthew, Sinha, Aman, Namkoong, Hongseok, Tedrake, Russ, and Duchi, John C.** *Scalable End-to-End Autonomous Vehicle Testing via Rare-event Simulation.* In: *Advances in Neural Information Processing Systems 31.* International Conference on Neural Information Processing Systems. Curran Associates, Inc., 2018, pp. 9827–9838.

[81] **Peharz, Robert**. *Foundations of sum-product networks for probabilistic modeling.* doctoralthesis. Graz: Graz University of Technology, 2015. 145 pp.

[82] **Poddey, Alexander, Brade, Tino, Stellet, Jan Erik, and Branz, Wolfgang**. *On the validation of complex systems operating in open contexts.* In: arXiv, 2019.

[83] **Poon, Hoifung and Domingos, Pedro**. *Sum-product networks: A new deep architecture.* In: 2011 IEEE International Conference on Computer Vision Workshops (ICCV Workshops). Barcelona, Spain: IEEE, 2011, pp. 689–690.

[84] **Rashwan, Abdullah, Poupart, Pascal, and Zhitang, Chen**. *Discriminative Training of Sum-Product Networks by Extended Baum-Welch.* In: *Proceedings of the Ninth International Conference on Probabilistic Graphical Models.* Ed. by **Kratochvíl, Václav and Studený, Milan**. Vol. 72. Proceedings of Machine Learning Research. Prague, Czech Republic: PMLR, 11, 2018, pp. 356–367.

[85] **Rivalin, Lisa, Stabat, Pascal, Marchio, Dominique, Caciolo, Marcello, and Hopquin, Frédéric**. *A comparison of methods for uncertainty and sensitivity analysis applied to the energy performance of new commercial buildings.* In: *Energy and Buildings* 166 (2018), pp. 489–504.

[86] **Rocklage, Elias**. *Teaching self-driving cars to dream: A deeply integrated, innovative approach for solving the autonomous vehicle validation problem.* In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). Yokohama, Japan: IEEE, 2017, pp. 463–469.

[87] **Rocklage, Elias, Kraft, Heiko, Karatas, Abdullah, and Seewig, Jorg**. *Automated scenario generation for regression testing of autonomous vehicles.* In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). Yokohama, Japan: IEEE, 2017, pp. 476–483.

[88] **Ruano, M. V., Ribes, J., Seco, A., and Ferrer, J.** *An improved sampling strategy based on trajectory design for application of the Morris method to systems with many input factors.* In: *Environmental Modelling & Software* 37 (2012), pp. 103–109.

[89] **Rubinstein, Reuven Y.** *The Cross-Entropy Method for Combinatorial and Continuous Optimization*. In: *Methodology And Computing In Applied Probability* 1 (1999), pp. 127–190.

[90] **Saltelli, Andrea, Tarantola, Stefano, Campolongo, Francesca, and Ratto, Marco**. *Sensitivity Analysis in Practice: A Guide to Assessing Scientific Models*. Hoboken, NJ: Wiley, 2004. 219 pp.

[91] **Sarkar, Atrisha and Czamecki, Krzysztof**. *A behavior driven approach for sampling rare event situations for autonomous vehicles*. In: 2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS). Macau, China: IEEE, 2019, pp. 6407–6414.

[92] *Schlussbericht für das Gesamtprojekt Pegasus*. 2020.

[93] **Schönebeck, Susanne, Schepers, Andreas, Pöppel-Decker, Martin, Färber, Nadja, and Fitschen, Arnd**. *Voraussichtliche Entwicklung von Unfallanzahlen und Jahresfahrleistungen in Deutschland -Ergebnisse 2019-*. Bundesanstalt für Straßenwesen (bast), 2019.

[94] **Schuldt, Fabian**. *Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen*. doctoralthesis. Technische Universität Braunschweig, 2017.

[95] **Schuldt, Fabian, Saust, Falko, Lichte, Bernd, Maurer, Markus, and Scholz, Stephan**. *Effiziente systematische Testgenerierung für Fahrerassistenzsysteme in virtuellen Umgebungen*. In: *Automatisierungssysteme, Assistenzsysteme und Eingebettete Systeme Für Transportmittel*. (2013).

[96] **Schwabl, Franz**. *Statistische Mechanik*. OCLC: 1229591302. Heidelberg: Springer-Verlag Berlin Heidelberg, 2006.

[97] **Shalev-Shwartz, Shai, Shammah, Shaked, and Shashua, Amnon**. *On a formal model of safe and scalable self-driving cars*. In: *arXiv preprint arXiv:1708.06374* (2017).

[98] **Stellet, Jan Erik, Zofka, Marc Rene, Schumacher, Jan, Schamm, Thomas, Niewels, Frank, and Zollner, J. Marius**. *Testing of Advanced Driver Assistance Towards Automated Driving: A Survey and Taxonomy on Existing Approaches and Open Questions*. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. Las Palmas, Spain: IEEE, 2015, pp. 1455–1462.

[99] **Strigel, Elias, Meissner, Daniel, Seeliger, Florian, Wilking, Benjamin, and Dietmayer, Klaus**. *The Ko-PER intersection laserscanner and video dataset*. In: 17th International IEEE Conference on Intelligent Transportation Systems (ITSC). Qingdao, China: IEEE, 2014, pp. 1900–1901.

[100] *SUMO User Documentation*. URL: https://sumo.dlr.de/docs/index.html (visited on 11/20/2020).

[101] **Sun, Jian, Zhou, Huajun, Zhang, He, Tian, Ye, and Qinghui, Ji**. *Adaptive Design of Experiments for Accelerated Safety Evaluation of Automated Vehicles*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). Rhodes, Greece (Virtual): IEEE, 2020, pp. 8–14.

[102] **Tan, Shuhan, Wong, Kelvin, Wang, Shenlong, Manivasagam, Sivabalan, Ren, Mengye, and Urtasun, Raquel**. *SceneGen: Learning to Generate Realistic Traffic Scenes*. In: 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 2021, pp. 892–901.

[103] **Tongeren, Robin van, Gietelink, Olaf, De Schutter, Bart, and Verhaegen, Michel**. *Traffic Modelling Validation of Advanced Driver Assistance Systems*. In: IEEE, 2007, pp. 1246–1251.

[104] **Treiber, Martin, Hennecke, Ansgar, and Helbing, Dirk**. *Congested traffic states in empirical observations and microscopic simulations*. In: *Physical Review E* 62.2 (1, 2000), pp. 1805–1824.

[105] **Tuncali, Cumhur Erkan and Fainekos, Georgios**. *Rapidly-exploring Random Trees for Testing Automated Vehicles*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). Auckland, New Zealand: IEEE, 2019, pp. 661–666.

[106] **Tuncali, Cumhur Erkan, Pavlic, Theodore P., and Fainekos, Georgios**. *Utilizing S-TaLiRo as an automatic test generation framework for autonomous vehicles*. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). Rio de Janeiro, Brazil: IEEE, 2016, pp. 1470–1475.

[107] **Tuncali, Cumhur Erkan, Fainekos, Georgios, Ito, Hisahiro, and Kapinski, James**. *Simulation-based Adversarial Test Generation for Autonomous Vehicles with Machine Learning Components*. In: 2018 IEEE Intelligent Vehicles Symposium (IV). Changshu, China: IEEE, 2018, pp. 1555–1562.

[108] **Uesato, Jonathan, Kumar, Ananya, Szepesvari, Csaba, Erez, Tom, Ruderman, Avraham, Anderson, Keith, Dvijotham, Krishmamurthy, Heess, Nicolas, and Kohli, Pushmeet**. *Rigorous Agent Evaluation: An Adversarial Approach to Uncover Catastrophic Failures*. In: *arXiv preprint arXiv:1812.01647* (2018), p. 20.

[109] **Ulbrich, Simon, Menzel, Till, Reschka, Andreas, Schuldt, Fabian, and Maurer, Markus**. *Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving*. In: IEEE 18th International Conference on Intelligent Transportation Systems. IEEE, 2015, pp. 982–988.

[110] *Verkehr - Verkehrsunfälle - 2019*. Fachserie 8 Reihe 7. Statistisches Bundesamt (Destatis), 2020.

[111] **Wachenfeld, Walter and Winner, Hermann**. *Virtual Assessment of Automation in Field Operation - A New Runtime Validation Method*. In: 10. Workshop Fahrerassistenzsysteme. Walting im Altmühltal, 2015.

[112] **Wachenfeld, Walther and Winner, Hermann**. *Die Freigabe des autonomen Fahrens*. In: *Autonomes Fahren*. Ed. by **Maurer, Markus, Gerdes, J. Christian, Lenz, Barbara, and Winner, Hermann**. DOI: 10.1007/978-3-662-45854-9_21. Berlin, Heidelberg: Springer Berlin Heidelberg, 2015, pp. 439–464.

[113] **Wachenfeld, Walther, Junietz, Philipp, Wenzel, Raphael, and Winner, Hermann**. *The worst-time-to-collision metric for situation identification*. In: 2016 IEEE Intelligent Vehicles Symposium (IV). Gothenburg, Sweden: IEEE, 2016, pp. 729–734.

[114] **Wang, Hongling, Kearney, Joseph, and Atkinson, Kendall**. *Robust and efficient computation of the closest point on a spline curve*. In: *Proceedings of the 5th International Conference on Curves and Surfaces*. 2002, pp. 397–406.

[115] **Wang, Wenshuo and Zhao, Ding**. *Evaluation of Lane Departure Correction Systems Using a Regenerative Stochastic Driver Model*. In: *IEEE Transactions on Intelligent Vehicles* 2.3 (2017), pp. 221–232.

[116] **Wang, Xinpeng, Peng, Huei, and Zhao, Ding**. *Combining Reachability Analysis and Importance Sampling for Accelerated Evaluation of Highly Automated Vehicles at Pedestrian Crossing*. In: ASME 2019 Dynamic Systems and Control Conference. Park City, Utah, USA: American Society of Mechanical Engineers, 8, 2019.

[117] **Wheeler, Tim A.** *Automotive safety validation in simulation*. doctoralthesis. Stanford University, 2018. 216 pp.

[118] **Wheeler, Tim A. and Kochenderfer, Mykel J.** *Factor graph scene distributions for automotive safety analysis*. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). Rio de Janeiro, Brazil: IEEE, 2016, pp. 1035–1040.

[119] **Wheeler, Tim A., Kochenderfer, Mykel J., and Robbel, Philipp**. *Initial Scene Configurations for Highway Traffic Propagation*. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems (ITSC). Las Palmas, Spain: IEEE, 2015, pp. 279–284.

[120] **Wheeler, Tim A., Robbel, Philipp, and Kochenderfer, Mykel J.** *Analysis of microscopic behavior models for probabilistic modeling of driver behavior*. In: 2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC). Rio de Janeiro, Brazil: IEEE, 2016, pp. 1604–1609.

[121] **Wheeler, Tim Allan and Kochenderfer, Mykel J.** *Critical Factor Graph Situation Clusters for Accelerated Automotive Safety Validation*. In: 2019 IEEE Intelligent Vehicles Symposium (IV). Paris, France: IEEE, 2019, pp. 2133–2139.

[122] **Wiedemann, Rainer**. *Simulation des Straßenverkehrsflusses*. In: *Institute for Transportation Science, University of Karlsruhe, Germany* Schriftenreihe Heft 8 (1974).

[123] **Wishart, Jeffrey, Como, Steven, Elli, Maria, Russo, Brendan, Weast, Jack, Altekar, Niraj, James, Emmanuel, and Chen, Yan**. *Driving Safety Performance Assessment Metrics for ADS-Equipped Vehicles*. In: WCX SAE World Congress Experience. SAE International, 2020, pp. 2881–2899.

[124] **Zhang, Songan, Peng, Huei, Zhao, Ding, and Tseng, H. Eric**. *Accelerated Evaluation of Autonomous Vehicles in the Lane Change Scenario Based on Subset Simulation Technique*. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC). Maui, HI, USA: IEEE, 2018, pp. 3935–3940.

[125] **Zhao, Ding, Guo, Yaohui, and Jia, Yunhan Jack**. *TrafficNet: An open naturalistic driving scenario library*. In: 2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC). Yokohama, Japan: IEEE, 2017, pp. 1–8.

[126] **Zhao, Ding, Lam, Henry, Peng, Huei, Bao, Shan, LeBlanc, David J., Nobukawa, Kazutoshi, and Pan, Christopher S.** *Accelerated Evaluation of Automated Vehicles Safety in Lane-Change Scenarios Based on Importance Sampling Techniques*. In: *IEEE Transactions on Intelligent Transportation Systems* 18.3 (2017), pp. 595–607.

[127] **Zhao, Ding, Huang, Xianan, Peng, Huei, Lam, Henry, and LeBlanc, David J.** *Accelerated Evaluation of Automated Vehicles in Car-Following Maneuvers*. In: *IEEE Transactions on Intelligent Transportation Systems* 19.3 (2018), pp. 733–744.

[128] **Zhou, Jinwei and Re, Luigi del**. *Identification of critical cases of ADAS safety by FOT based parameterization of a catalogue*. In: 2017 11th Asian Control Conference (ASCC). Gold Coast, QLD, Australia: IEEE, 2017, pp. 453–458.

[129] **Zhou, Jinwei and Re, Luigi del**. *Reduced Complexity Safety Testing for ADAS & ADF*. In: *IFAC-PapersOnLine* 50.1 (2017), pp. 5985–5990.

[130] **Zhou, Jinwei and Re, Luigi del**. *Safety Verification Of ADAS By Collision-free Boundary Searching Of A Parameterized Catalog*. In: 2018 Annual American Control Conference (ACC). Milwaukee, WI, USA: IEEE, 2018, pp. 4790–4795.

[131] **Zofka, Marc René, Kuhnt, Florian, Kohlhaas, Ralf, Rist, Christoph, Schamm, Thomas, and Zöllner, J. Marius**. *Data-driven simulation and parametrization of traffic scenarios for the development of advanced driver assistance systems*. In: 2015 18th International Conference on Information Fusion (Fusion). Washington, DC, USA, 2015.

List of Publications

[132] **Jesenski, Stefan, Stellet, Jan Erik, Schiegg, Florian, and Zollner, J. Marius**. *Generation of Scenes in Intersections for the Validation of Highly Automated Driving Functions*. In: 2019 IEEE Intelligent Vehicles Symposium (IV). © 2019 IEEE. Reprinted, with permission. Paris, France: IEEE, 2019, pp. 502–509.

[133] **Jesenski, Stefan, Stellet, Jan Erik, Branz, Wolfgang, and Zöllner, J. Marius**. *Simulation-Based Methods for Validation of Automated Driving: A Model-Based Analysis and an Overview about Methods for Implementation*. In: 2019 IEEE Intelligent Transportation Systems Conference (ITSC). © 2019 IEEE. Reprinted, with permission. Auckland, New Zealand: IEEE, 2019, pp. 1914–1921.

[134] **Jesenski, Stefan, Tiemann, Nils, Stellet, Jan Erik, and Zöllner, J. Marius**. *Scalable Generation of Statistical Evidence for the Safety of Automated Vehicles by the Use of Importance Sampling*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). © 2020 IEEE. Reprinted, with permission. Rhodes, Greece (Virtual): IEEE, 2020, pp. 1240–1247.

[135] **Jesenski, Stefan, Rothert, Jakob, Tiemann, Nils, and Zöllner, J. Marius**. *Using Sum-Product Networks for the Generation of Vehicle Populations On Highway Sections*. In: 2020 IEEE Intelligent Transportation Systems Conference (ITSC). © 2020 IEEE. Reprinted, with permission. Rhodes, Greece (Virtual): IEEE, 2020, pp. 1883–1889.

[136] **Jesenski, Stefan, Tiemann, Nils, Branz, Wolfgang, and Zöllner, J. Marius**. *Creation of Critical Traffic Scenes for Usage with Importance Sampling*. In: 2021 IEEE International Conference on Intelligent Transportation (ITSC). © 2021 IEEE. Reprinted, with permission. Indianapolis, USA: IEEE, 2021, pp. 3162–3169.

[137] **Schiegg, Florian A., Krost, Johannes, Jesenski, Stefan, and Frye, Johannes**. *A Novel Simulation Framework for the Design and Testing of Advanced Driver Assistance Systems*. In: 2019 IEEE 90th Vehicular Technology Conference (VTC2019-Fall). Honolulu, HI, USA: IEEE, 2019, pp. 1–6.

[138] **Stellet, Jan Erik, Brade, Tino, Poddey, Alexander, Jesenski, Stefan, and Branz, Wolfgang**. *Formalisation and algorithmic approach to the automated driving validation problem.* In: 2019 IEEE Intelligent Vehicles Symposium (IV). Paris, France: IEEE, 2019, pp. 45–51.