Scene to Scenario: Data-driven Pipeline for Extracting and Re-Simulation of Test Scenarios for Highly Automated Driving Functions

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

M.Sc. Maximilian Zipfl

Tag der mündlichen Prüfung:	05.03.2025
Referent:	Prof. Dr. J. Marius Zöllner
Korreferent:	Prof. Dr.-Ing. Ina Schaefer
Karlsruhe	2025

Acknowledgments and Preface

I would like to thank everyone who has supported me along the way! My thanks go to colleagues, friends and family, whose support and encouragement made this work possible in the first place. To begin with, I would like to thank Prof. Dr.-Ing. J. Marius Zöllner for his continuous scientific supervision, the constructive discussions and many important comments and, of course, undertaking the main supervision. Furthermore, I would like to thank Prof. Dr.-Ing. Ina Schaefer for being the co-supervisor and the helpful comments on my work.

I would also like to thank all my colleagues at the FZI in the TKS department for the excellent working atmosphere and the many productive discussions. Special thanks go to Daniel Bogdoll for the countless discussions on research topics and the resulting ideas, valuable comments and proofreading of this thesis, which contributed significantly to the quality of this work. I would also like to thank Barbara Schütt for the many debates on criticality metrics and the assurance of highly automated vehicles. And of course also for the co-authored publications, which also contributed significantly to this work. A big thank-you also goes to all the students who have supported me in my research in the context of scientific assistant activities, Bachelor's or Master's theses. A special thank-you goes to Sven Spickermann and Moritz Jarosch.

I would also like to thank the consortium of the VVMethoden project for the many technical discussions on highly automated driving. A large part of the research presented in this thesis was developed in this context.

A huge thank-you goes to my friends and family, especially my parents. They were always there for me and supported me wherever they could. Last but not least, I would like to say a special thank you to Hella. Without her years of support and patience for the many evenings and weekends I spent working on this dissertation, I would not have been able to complete this work - Thank you.

Highly automated driving will play an important role in the near future, at least in some areas of mobility. At this point, user confidence in this new technology will play a decisive role. An independent test procedure that tests such complexes is therefore an important component for the widespread use of highly automated driving systems. In this dissertation, I would like to highlight a small part of such a testing procedure and, more specifically, present an approach that can facilitate the testing of highly automated driving functions. The methods presented attempt to close existing research gaps and have been tested and evaluated using prototypical implementations. In particular, the Python programming language and corresponding libraries (PyTorch, PyTorch Geometric) were used for the application of machine learning and the implementation of software within this work. While writing this thesis, tools like the DeepL Translator, DeepL Write, LanguageTool, and other GPT-based tools were used to proofread the language and improve the grammar.

Karlsruhe, 2024 *Maximilian Zipfl*

Abstract

The development of highly automated driving has made remarkable progress in recent years. Numerous technical challenges have been solved, allowing the first partly autonomous vehicles to drive on public roads. However, the safeguarding process of Highly Automated Vehicles (HAVs) remains a significant challenge. The verification and validation of a corresponding system play a crucial role for manufacturers, as they ensure that their own systems are free of faults. Furthermore, verification carried out by testing institutions is relevant in order to allow new systems on public roads without posing a safety risk to traffic participants. The highly complex nature of HAVs makes validation utilizing statistical tests considerably more difficult. In this context, scenario-based testing in a simulation opens up promising perspectives for the validation.

This dissertation presents and discusses a concept for the creation and evaluation of test scenarios for Highly Automated Driving Functions (HADFs). An essential aspect in the creation of a test scenario is the mutual consideration of the behavior of all traffic participants within the simulation. The trajectory and solution strategies may vary depending on the HADF being tested. Consequently, the other simulated agents must react consistently. To remain as comprehensive as possible, the dissertation does not consider any specific HADF and its behavior. Rather, this thesis deals in particular with the initial configuration, i.e. the traffic scene, which serves as the starting point for conducting a subsequent test.

A description model, referred to as the Semantic Scene Graph, is presented. This model allows the description of individual traffic scenes based on the constellations of traffic participants and relationships between them. The model facilitates a direct comparison between traffic scenes and serves as a basis for following modules. In the next step, the abstract description model is used to check scenes for similarity using a graph-based self-supervised machine learning method. The result is a structured scene space that can be organized into clusters of similar traffic scenes.

After the scenes have been grouped into similar categories, they have to be evaluated based on their relevance for testing. A number of different criticality metrics serve as an objective measure to evaluate the criticality of a situation. In order to make a meaningful statement about the relevance of an initial scene (seed-scene), many different possible futures are simulated that can develop from a seed-scene. In addition to rule-based methods with varying parameterizations and combinations, two machine-learned behavior models developed in this thesis are used to simulate a broad spectrum of potential futures for a seed-scene. The resulting child-scenarios are then evaluated using the criticality metrics. A subsequent analysis of the criticality distribution provides insight into how far and in what aspects a seed-scene can be relevant for the generation of a test scenario. The criticality profile, in combination with the information pertaining to the associated cluster, can be utilized to identify traffic scenes that are characterized by important attributes.

In summary, this dissertation presents an approach for identifying important scenes from real-world data that can be used for testing Highly Automated Driving Functions. Within this pipeline, traffic scenes are evaluated based on their criticality and organized into clusters of similar scenes within the scene space.

Contents

Glossary

ADE	Average Displacement Error . 90, 91
FDE	Final Displacement Error . 90, 91
ADAM	Adaptive Moment Estimation. 68, 72, 89
CNN	Convolutional Neural Network. 18, 24, 81, 82
CP	Criticality Potential. 105–111, 122, 125, 153–159, 162, 165, 166
Dist	Euclidean Distance. 49, 52, 53, 55–57, 102, 105, 107, 110, 120, 122, 123, 153, 162, 165, 167
DTW	Dynamic Time Warping. 60, 61
ET	Encroachment-Time. 51, 53, 55–57, 120
GAT	Graph Attention Network. 82
GCN	Graph Convolutional Network. 82
GNN	Graph Neural Network. 8, 18, 19, 31, 59, 62, 63, 65, 76, 82–85, 88
GRU	Gated Recurrent Network. 81, 87, 88
GT	Gap Time. 51, 53, 55–57, 102, 105–107, 110, 120, 122, 155, 162, 165
HADF	Highly Automated Driving Function. iii, iv, 1, 4–6, 8, 95, 97, 109, 113–115
HAV	Highly Automated Vehicle. iii, 1–4, 13–15, 17, 59, 95, 96, 98, 119
IDM	Intelligent Driver Model. 80, 92, 93, 96, 97, 100
KDE	Kernel Density Estimation. 102
Leaky ReLU	Leaky Rectified Linear Unit. 68, 85, 88

LIDAR Light Detection and Ranging. 24

LSTM Long Short-term Memory. 81, 83, 85, 86

MAE Mean Absolute Error. 72, 73

MLP Multilayer Perceptron. 19, 62, 66, 68, 72, 85, 86, 88

MSE Mean Squared Error. 72, 73

ODD Operational Design Domain. 1, 2, 11, 12, 14, 15, 47

PCA Principal Component Analysis. 61, 70, 71, 120

PET Post-encroachment-time. 49, 51, 53, 55–57, 107, 120, 122, 156, 162, 165

PTTC Potential Time To Collision. 50, 53, 56, 57, 102, 107, 122, 157, 162

RDF Resource Description Framework. 27, 28

RNN Recurrent Neural Network. 81, 82

RSS Responsibility Sensitive Safety. 11, 52

SAE Society of Automotive Engineers. 1

SFF Safety Force Field. 2, 11, 52

SOTIF Safety of The Intended Functionality. 9

SP Safety potential. 52, 53, 55–57, 120

SSG Semantic Scene Graph. iii, xiv, 30–32, 34, 37, 38, 42–45, 59, 62, 64, 79, 84–86, 90, 91, 114–116, 120

Std Standard Deviation. 72, 73

SuT System under Test. 3, 4, 10, 12, 47, 59, 95, 96, 113, 117

t-SNE t-distributed Stochastic Neighbor Embedding. 107

TD Trajectory Distance. 49, 53, 55–57, 120

TQ Inverse Universal Traffic Quality. 52, 53, 55–57, 102, 105, 107, 110, 120, 122, 123, 154, 162, 165, 167

TTC Time To Collision. 49–53, 55–57, 102, 120, 122, 158, 162

TTC* inverse Time To Collision. 50, 102, 105, 107, 110, 122, 123, 165, 167

UMAP Uniform Manifold Approximation and Projection. 107

V2X Vehicle-To-Everything. 26

WTTC Worst Time To Collision. 50, 51, 53, 55–57, 102, 107, 120, 122, 159, 162

Symbols

a Acceleration of a traffic participant. 50, 92

a_b Comfortable braking acceleration of the IDM. 92, 102

$\hat{a}$ Acceleration target which is to be predicted by a model. 85, 89, 90

a_{max} Maximum possible acceleration of the IDM. 92

$\tilde{a}$ Acceleration target which is to be predicted of a model. 84, 89, 90

α Factor determining how much a particular embedding is relied upon to predict the hybrid trajectory. 88, 121

α' Sensitivity coefficient to scale the value range of a criticality metric. 53

β Angular deviation between two vectors. 93

$\mathcal{D}$ Dimensionality of the embedding space of the graph neural network encoder. 67, 68, 70

d Distance metric for comparing two samples in the embedding space. 67, 68, 71, 72

d_F Frenet distance; distance along the lane. 35–38, 40, 49, 50, 92, 102

d_{ip} Distance along the lane to the intersection point. 35–37, 40

d_n Euclidean distance between the center line of a road and a traffic participant. 33–36, 38, 40, 93

ε Margin of error. 103, 104

ϵ Latent feature representation (embedding). 85–88, 121

E_{road} Set of all edges between elementary road segments. 31, 38

e_{road} Edge describing, how two elementary road segments are connected. 31, 36

E_{scene} Set of all edges between elementary projected traffic participants in the scene graph. 34

e_{scene} Edge describing, how two projected traffic participants are connected. 35–40

G General graph description for processing derived from the SSG. 85, 87, 88, 121

G_{-} Negative graph sample of the graph triplet. 63, 66, 67, 120

G_0 Anchor graph sample of the graph triplet. 62–64, 66, 67, 120

G_{+} Positive graph sample of the graph triplet. 62–64, 66, 67, 120

G_{road} Directed road graph which holds the topological information of the road network. 31, 34–36, 38

G_{scene} Directed scene graph (Semantic Scene Graph) which holds the topological information of traffic participants in regard to the road topology. 34, 38, 39, 43

γ Update function, which updates the node's state based on the current state of the node and the aggregated messages. 19, 65, 66, 68

h Embedding of a node in a graph. 19, 20, 65, 66, 84, 85, 87, 88

I Set of all traffic participants in the scene. 22, 34, 38, 41

I' Subset of all traffic participants which can be projected to a road segment. 38

$\mathcal{I}$ Top-down image representation. 87, 88, 121

κ_{obj} Classification of the traffic participant (pedestrian, vehicle, ...). 22, 39, 40

κ_{road} Defining the type, on how road segments are connected (Consecutive, Adjacent, Overlapping). 31, 37

κ_{rel} Defining the type, on how traffic participants are connected (Following, Adjacent, Intersecting). 34–40

$\mathcal{L}$ Loss function to train a neural network. 89

$\mathcal{L}_{triplet}$ Triplet loss function. 67, 68

l Length of a traffic participant. 22

M Triplet margin used to distance negative samples by a desired spread in the triplet loss. 67, 68

m A projection identity describes probable poses of a traffic participant on nearby road segments. 33–36, 38–41, 119

m Message function, which processes incoming message information from neighboring nodes. 19, 20, 65–68

MP Message-passing function, which describes how information is passed from one neighboring node to another. 19, 65–67

ν Velocity of a traffic participant. 49, 50, 73, 84, 92

ν_0 Target velocity for the IDM. 92

$\mathfrak{p}$ Point in geometric space. 37, 38, 93

P_{entity} Specifies the probability with which a traffic participant is selected as to whether its state is modified. 63, 68

P_{proj} Probability of the projection assignment between traffic participant and road segment defined. 33, 35

P_{rel} Probability of an edge between projection identities. 35, 40

$\tilde{P}_{rel}$ Normalized probability of an edge type over all other probabilities between nodes. 40

$\mathcal{P}_{road}$ Set of path in the road graph. 38, 39

p_{road} Path along vertices in the road graph. 35, 36

Φ Angular deviation between a road segments' centerline and the orientation of the traffic participant. 33–35, 37, 40

ϕ Multilayer perceptron projection head with intermediary activation function. 65, 66, 84–86, 88

ψ Orientation (yaw) of a local coordinate system. 22, 38, 93, 121

R Readout function for aggregating embeddings of multiple nodes in a graph. 65, 66, 68

S Embedding space. 62, 66–68, 70–72, 74, 77, 120, 121

$S_{\mathbb{R}^2}$ State of a traffic scene in the Cartesian space at a given time step, containing traffic participants and the road data. 22, 33, 38, 43, 62–64, 99, 120, 121

s_- Graph embedding of the negative sample of the graph triplet. 62, 63, 66, 67, 71, 72

s_0 Graph embedding of the anchor sample of the graph triplet. 62, 63, 66, 67, 71, 72

s_+ Graph embedding of the positive sample of the graph triplet. 62, 63, 66, 67, 71, 72

σ_d Standard deviation to weight the distance of the road segments considered. 33

σ_p Standard deviations to weight the orientation of the road segments considered. 33

σ_{pos} Standard deviation used to parameterize the range of values for modifying the position of a traffic participant. 63, 68

σ_{vel} Standard deviation used to parameterize the range of values for modifying the velocity of a traffic participant. 63, 68

$\hat{\mathcal{T}}$ Trajectory predicted by a model. 87–89, 99

$\tilde{\mathcal{T}}$ Trajectory target which is to be predicted from a model. 89

T_0 Time headway of the IDM. 102

$\hat{T}$ Length of a future timestep sequence. 88, 90, 101, 102, 121

$\tilde{T}$ Length of the history for a timestep sequence. 85–88

Θ Learnable weight matrix. 84, 85, 87

V_{road} Set of all elementary road segments. 31, 41

v_{road} Node which describes a elementary road segment. 31, 35, 36

V_{scene} Set of all projected traffic participants represented by nodes. 34

v_{scene} Node which describes a elementary a projected traffic participant in the scene graph. 35, 36, 39, 65

w Width of a traffic participant. 22

x Geometric center of the traffic participant on the x-axis of a local coordinate system. 22, 38, 63, 88

$\dot{x}$ Speed of a traffic participant in the direction of the x-axis of a local coordinate system. 22, 40, 63

$\ddot{x}$ Acceleration of the traffic participant in the direction of the x-axis of a local coordinate system. 22

$\hat{\mathcal{X}}$ State of a traffic participant and its projection identities after projection onto the road network. 34, 38

$\mathcal{X}$ State of a traffic participant for a given timestep t. 22, 38, 63

y Geometric center of the traffic participant on the y-axis of a local coordinate system. 22, 38, 63, 88

$\dot{y}$ Speed of a traffic participant in the direction of the y-axis of a local coordinate system. 22, 40, 63

$\ddot{y}$ Acceleration of the traffic participant in the direction of the y-axis of a local coordinate system. 22

z Standard score for the sample size estimation method. 103, 104

1 Introduction

In recent years, highly automated driving is considered one of the most important strategic research areas in the automotive industry and related sectors [27]. As vehicles become more intelligent, they promise to reduce the amount of tasks performed by the driver and make driving more relaxed, while at the same time improving driving safety and reducing the number of road accidents.

In the meantime, the Society of Automotive Engineers (SAE) International has established the six levels of vehicle automation as a descriptive standard [28]. In principle, this standard describes the capability of the system, the associated Operational Design Domain (ODD), i.e. the space with all the scenarios the system is designed for, and whether the fallback level is the human or the system. Starting from SAE Level 0, where the system lacks any automation. SAE Level 1 and 2 indicate systems that can aid the driver in specific driving tasks, such as lane departure warning systems or distance-keeping assistants. From SAE Level 3, the vehicle is fully controlled by the system in certain areas and is referred to as a Highly Automated Vehicle (HAV). At SAE Level 4, the human takes a secondary role and the system is able to drive completely autonomously in a predefined ODD. SAE Level 5, which represents the final level, refers to systems that can independently handle all possible situations, resulting in the largest possible ODD.

The most prominent example, and possibly the most advanced Highly Automated Driving Function (HADF) available to the public at the moment, is the Waymo Driver from the company Waymo. Waymo's robotaxi can operate completely autonomously in a geographically limited area and does not require a safety driver, meaning that the Waymo Driver falls into the SAE Level 4 autonomy category [29, 30]. The first vehicle (of Mercedes) with a SAE Level 3 traffic jam pilot is already permitted to the road in Germany. Compared to the Waymo vehicle, there are further restrictions in the ODD, which means that the system may be used up to a maximum speed of $60\,\frac{\mathrm{km}}{\mathrm{h}}$ and not in the rain, for example. Despite the enormous progress made in recent years in the research and development of HAVs, researchers in the field agree that the development of autonomous driving is far from mature [31, 32, 27]. There are still several issues that must be resolved for the world-wide implementation of autonomous vehicles. These challenges include legal matters, necessitating collaboration between the industry and politics to establish regulations and legislation that promote self-driving but also establish boundaries. The design issues concentrate on technical solutions for a reliable understanding of the situation and consequent driving decisions. In this context, it is necessary to analyze not only the individual components of the system, such as sensors, hardware components, and subsystems like perception, but also how the entire system functions collectively.

Ensuring the correct functionality of the vehicle and guaranteeing the greatest possible level of safety are of key interest, a quantitative assessment is necessary [33]. The authors of the Safety Force Field (SFF)-Model [34] suggest an acceptable error rate of $10^{-9}\,\frac{1}{\mathrm{h}}$ for HAV. The authors Wachenfeld and Winner [35] suggest that HAV require a test distance of 2.1 billion kilometers for confirmation, using comparable reasoning. The distances and durations for verification mentioned here reveal a distinct trend: **statistical test methods are not economically compatible for the validation of automated vehicles, indicating a need for alternative verification strategies.** This necessitates a comprehensive adaptation of previous safety methods and homologation[*] processes of vehicles that were previously operated by humans [35, 36, 37].

In recent years, scenario-based testing methods have been the subject of investigation. The creation of artificial scenarios that lie within the ODD is a specific focus of these approaches, which is necessary for the approval process. In 2022, the European Commission passed the Implementing Regulation (EU) 2022/1426 [38]. This regulation provides specifications at the European level for assessing the safety of an HAV and for the approval of these systems. The legal text demands that the manufacturer must prove that the functional and operational safety of the HAV has been taken into account in the development process. To this end, it is necessary to define specific acceptance criteria. In order to demonstrate the safety of their system, the company has to document the tests on critical scenarios. Consequently, these critical scenarios then have to be used on a random basis by a testing institution to verify reliability. Simulation frameworks should be used for testing, in particular for scenarios that are difficult to re-enact in reality.

In general, the regulation specifies functional test scenarios. However, there is no concrete definition of what the test catalog should look like, as this can vary depending on the ODD or system architecture. Due to the relatively recent regulation, hardly any vehicles have been approved in this context [39]. In the USA, where several approvals have already taken place, companies do not have to make their entire safeguarding process available to the public, resulting in the lack of knowledge concerning the utilized test process [40, 41]. Consequently, there is no consensus regarding the precise nature of the

[*]approval that allows a vehicle to be released to a market

safeguarding process, particularly in regard to the specifics of test scenarios. This is the starting point of this work. In the following, issues related to the motivated scenario generation for the safety of automated driving functions will be elaborated and discussed.

1.1 Research Objective and Topic Scope

It is reasonable to assume that scenario-based tests substantially decrease the effort required by traditional test approaches (see Section 2.1.4). However, by using scenario-based testing as a validation technique for HAV components, a variety of unanswered questions remain. Classically, the goal of scenario-based testing is to collect relevant scenarios for a specific System under Test (SuT) and its requirements. Therefore, it is important to create and test scenarios that either provide instructive insights into the SuT or challenge the system in its functionality. This implies that a procedure must be developed to describe the scenarios, thereby enabling the selection of customized scenarios for an SuT [42, 43]. This is employed, for instance, in scenario exploration to identify scenarios of particular criticality for a specific system [44, 45, 46].

In principle, three distinct methodologies exist for defining scenarios: data-driven approaches, knowledge-driven approaches and adversarial scenario generation [47] (for a more detailed discussion, please refer to Section 2.1.4). The comprehensive survey by Ding et al. [47] discusses the latest approaches to traffic scenario generation with respect to the aforementioned methodological groups. Among other things, the authors identify three important and still unsolved challenges: When generating scenarios for testing a system, it is appropriate to limit oneself to realistic scenarios. In terms of **fidelity**, it is desirable to consider only those scenarios that could occur in the real world. Scenarios that can be generated by parameterization but violate the laws of physics, are not relevant and should be neglected. But more importantly, for example, the behavior of traffic participants is variable, but still resembles a natural, human-like pattern. The second challenge is the **efficiency** of testing. Since critical scenarios are particularly rare, which makes statistical testing as described above very difficult, methods are needed that specifically generate critical scenarios. In many frameworks, tests are generated for specific SuTs. However, it should be possible to test different HAVs in general without the need of having to design a new test methodology for each different system and achieve **transferability** of said testing methods. The overall goal is to provide a generalized test framework that will, in the long run, be able to test HADFs from different vehicle manufacturers with little parameterization effort.

A prerequisite for the development of a test framework is a general understanding of the structure and components of the test subject. The following section provides a brief overview of the architecture of an HADF as a SuT and defines where the tests apply. Figure 1.1 shows a very simplified structure of a driving pipeline of an HAV. In principle, most automatic actions of an HAV start with a physical sensor that converts certain aspects of the environment into digital information. Perception algorithms are used to transform this raw data into a representation of the environment. Based on the abstract representation, the decision-making and planning module estimates the intentions of other dynamic traffic participants and makes decisions based on them and on the static environment. Calculated trajectories are then converted into steering and acceleration signals for the actuators.

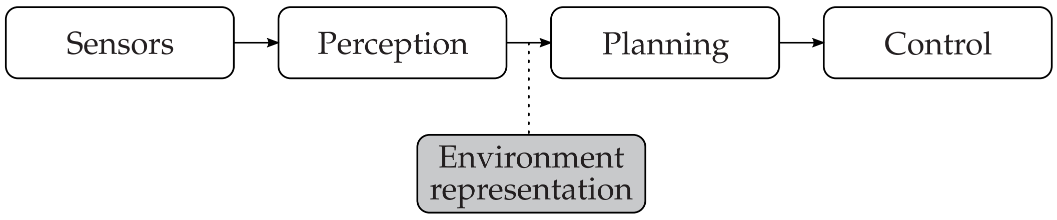


Figure 1.1: Structure of the driving pipeline of an HAV

This work's scope begins at the interface between the perception and planning modules. The representation of the environment is taken as given and interpreted without any noise (or uncertainty). The work is based on real data, which is further analyzed in a simulation. Although scenarios or scenes are evaluated for testing, as these are to be evaluated independently of specific driving functions, no driving functions are integrated in this work. Accordingly, the implementation on a real vehicle is not discussed, so neither the validation methods of the full autonomous vehicle nor the system components leading to the perception will be taken into account. Additionally, only the safety aspect and not the security aspect will be examined, therefore no intrusions into the system by malicious parties will be assessed.

Once the framework conditions have been defined, the following section will examine the challenges of scenario-based testing in more detail and derive research objectives. Methods for creating scenarios, such as exploration but especially exploitation-based methods, tend to focus on a specific SuT. In particular, optimization-based approaches generate especially critical scenarios very efficiently, but are usually only configured for one system [48, 49]. However, if an independent organization wants to test the driving functions of different manufacturers in an unbiased manner, a generalizable test method is advantageous. Therefore, the nuPlan benchmark [50] uses a closed-loop approach. In the majority of cases, there are several potential solutions to a given situation. Consequently, the application of different planning algorithms will result in the generation of alternative trajectories. In order to maintain consistency and accuracy, the environment has to respond accordingly [50, 51]. This leads to the first and overall Research goal 1:

> **Research goal 1.** What does a pipeline look like that generates test scenarios for the Highly Automated Driving Function under test as a black box?

Since the behavior of each scenario varies dynamically, the only constant is the initialization. Any change in the test-vehicle will lead to a completely different scenario development. Consequently, the initial step would be to generate a traffic scene (refer to Definition 1). At the same time, it is important to achieve high fidelity in order to avoid unrealistic scenes [47]. A major advantage of data-based scenario generation approaches is real data as a basis, which is inherently very close to reality [52]. An environment description serves as a simplified abstraction. It neglects unnecessary information, while still including all information relevant for subsequent tasks [53]. When selecting a description, it is important to ensure that the potential future of a traffic situation is accurately reflected. Consequently, the relationships between the various traffic participants are of significant importance. Furthermore, it is important to consider a wide variety of traffic scenes, regardless of the location where they were recorded. No two road geometries are the same, and the same situations can occur in different places without the road geometry having a direct influence on them. This raises the next research question (Research goal 2).

> **Research goal 2.** How can traffic scenes be described independently of the underlying road geometry?

The fundamental concept of scenario-based testing is that the scenario space can be divided into distinct types [43]. A cluster is considered tested when a few representative examples of this cluster have been analyzed. There are a number of different data-driven approaches to clustering traffic scenarios [54, 55, 56, 57]. However, in the majority of cases, these approaches are not applicable to the traffic scenes. Accordingly, the following Research goal 3 arises:

> **Research goal 3.** How to automatically cluster traffic scenes in a data-driven manner in terms of their similarity?

Only a tiny proportion of all scenarios represent a challenge for an HADF [58]. Accordingly, it is essential to reduce the test space to significant ones. This requires an independent assessment in order to prioritize the traffic scenarios according to their importance. A number of different criticality metrics can be employed to define specific aspects of a situation. However, it is often challenging for a single metric to provide a universal statement about the entirety of a scene or to differentiate between different aspects of different situation types [59]. To this end, a suitable evaluation method has to be found that ranks scenes according to their relevance (see Research goal 4):

> **Research goal 4.** How can a traffic scene be evaluated in a generalizable way to indicate its relevance to the test of a Highly Automated Driving Function?

In order to obtain the scenarios required in Research goal 1, the temporal development has to be defined. The scenario is based on a relevant scene (see Research goal 4). The fidelity demanded by Ding et al. [47] also plays an important role here, so that only realistic scenarios are generated. In order to bring the scenarios into a simulation, the various traffic participants must exhibit reactive behavior [50, 60]. In summary, Research goal 5 results from the fidelity and reactivity requirements for the behavior models.

> **Research goal 5.** How does the behavior of dynamic traffic participants have to be defined in the simulation to accurately represent the scenarios for testing?

The main focus of the planning module is the behavior of dynamic traffic participants. Consequently, different behaviors are reflected in different driving or moving trajectories. Other behaviors, such as head and limb movements, which can be relevant for vulnerable road users (pedestrians, cyclists), for example, are not considered in this scope.

1.2 Concept and Structure

In order to systematically address the research questions identified in Section 1.1, this thesis is divided into several chapters, with each chapter focusing primarily on one of the research goals. Methodological chapters are self-contained and provide related work and basic background information on the respective topic. However, it is recommended to read the chapters - and thus the entire work - in numerical order in order to put the related parts into context. Figure 1.2 shows the abstract structure of this thesis, which illustrates the dependencies of individual chapters and the systemic structure of the pipeline to answer Research goal 1.

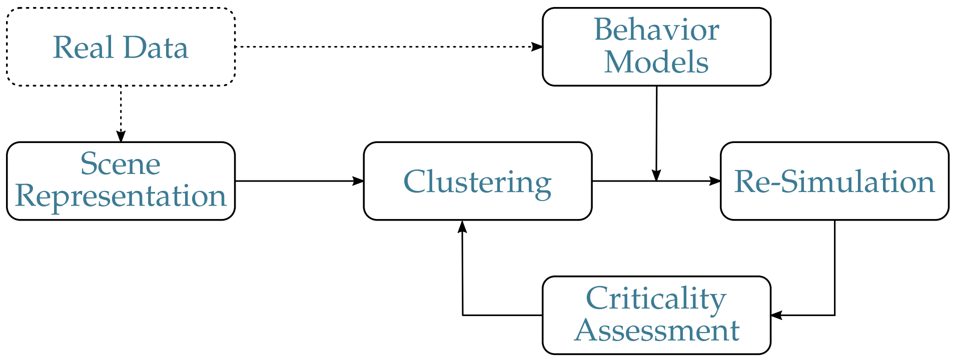


Figure 1.2: Thematic structure of the work and chapters and how they relate to one another in the larger context

The content of the methodological chapters has in many cases already been published, sometimes verbatim, in my previous works [1, 2, 3, 4, 5, 6, 7, 8, 9]. At the beginning of each chapter, a reference is made to the respective publications that are considered within the chapter.

The scene representation forms the first block of the methodology and is examined in more detail in Chapter 3. In order to give a special expression to the relationship between the individual traffic participants, the traffic scene is mapped onto a graph model. This graph forms the core of the further processing, abstracting real traffic into a machine-readable format, which enables the automated processing. The primary contribution of this chapter is the scene description model and how traffic scenes can be described in an abstract manner. This was previously published in work [1], and in full depth in work [8]. This chapter will therefore address Research goal 2. Subsequently, in Chapter 5, the traffic scenes described by the graphs are assigned to groups containing similar traffic scenes using a self-supervised machine learning process. The assignment and similarity determination of graphs represent a significant challenge. In this context, the previously published papers [9] and [4] present a graph representation learning approach that enables clustering in the latent state space.

In the next stage of the process, a generalizable statement is to be made about the grouped traffic scenes. To achieve this, a number of different potential futures that could arise from the scenes are to be simulated. The methodology used for the simulation and the subsequent analysis of the resulting scenarios are discussed in Chapter 7.

The evaluation of traffic scenarios is significantly influenced by the use of criticality metrics. These are described in detail in Chapter 4, and their general applicability is also examined. The key contribution of this chapter is the assessment of a traffic scene using a broad range of criticality metrics. The framework and its associated discussion, which facilitate criticality analysis, were presented in work [6]. The answer to Research goal 4 is provided in both Chapter 4 and in combination with the resulting evaluation methodology in Chapter 7. The contributions in the latter chapter concentrate on the assessment of a substantial number of simulated scenarios and the drawing of conclusions regarding the initial scene used. The simulation framework, the implementation, and the results were published in publication [3].

The selection of appropriate driver behavior models is a crucial component for the realization of a realistic simulation (see Research goal 5). Chapter 6 presents two trajectory predictors that employ imitation learning to mimic realistic vehicle trajectories. The behavior models based on a graph representation, which include explicit relations with other traffic participants, have been published in papers [7] and [5], respectively. These models can then be integrated into the simulation (see Chapter 7) to extrapolate the future of a scene.

In addition to the core methodology, Chapter 2 explains the basis for the verification and validation of autonomous driving and highlights both scenario-based testing and alternatives. In doing so, terms that are important for this work are defined, explained and the connection of the topics examined in this work is highlighted. These definitions serve as a basis and especially as a framework for the further chapters of the thesis. In addition, the fundamental principles of Graph Neural Networks (GNNs), including the message passing technique forming the basis of its operation, are presented. Finally, Chapter 8 summarizes the work, highlights the key findings, and identifies potential future research directions.

This thesis does not provide a definitive solution for scenario-based testing and therefore cannot provide absolute proof of the safety (see Section 2.1.1) of HADFs. Rather, it is aimed at demonstrating a methodology for identifying relevant test scenarios that can contribute to scenario-based testing.

2 Fundamentals

2.1 Validation & Verification

As stated in the introduction, extensive validation of the highly automated vehicle is essential. In this chapter, the fundamentals of the definition of safety within the context of Functional Safety are explored, with a particular emphasis on Safety of The Intended Functionality (SOTIF) [61]. Additionally, it describes the most prominent validation techniques that facilitate testing of a vehicle or certain submodules in relation to safety.

2.1.1 Safety Definition

In the automotive industry, the process of ensuring safety-critical systems currently follows a process whose goal is to ensure that technical failures due to errors in software or electronic hardware components do not lead to safety-critical situations. This target behavior is outlined in ISO 26262 [62] as Functional Safety. Functional Safety depends on the correct functioning of the safety-related system. In this regard, the potential failure of the function is assessed and measures to prevent systematic failures are implemented to limit the residual risk, including random failures, to an acceptable level. Systemic failures always adversely impact the system in a similar manner and are caused by system development, such as incomplete safety requirement implementation. Random or statistical errors occur unpredictably during the system's operation [63].

The SOTIF, as defined by the ISO 21448 standard [61], serves as a supplementary standard that outlines how to effectively address safety risks that could arise without causing system failure. The objective of SOTIF is to enhance the understanding of the system's potential behavior, even in unanticipated application scenarios. If a system can be transferred to an unsafe state despite meeting all previously specified requirements, there is a safety hazard for the target function. The aim of SOTIF is to define a structured design process that prevents safety violations caused by a faulty target function [63].

According to the ISO 21448 [61], there are four categories of scenarios that can be classified as either known (I, II) or unknown (III, IV) and as hazardous (II, III) or non-hazardous (I, IV) (see Figure 2.1). Hazard scenarios are those that could potentially cause harm. Meanwhile, there are scenarios where system behavior remains to be identified or are completely unknown. The primary objective of safety assessment approaches is to evaluate the potentially dangerous behavior of hazard scenarios (II, III) and to provide an argument that the residual risk is below the acceptance criteria [61]. There are various techniques to identify unknown scenarios (III, IV) and convert them into known scenarios, thereby minimizing their number. Conversely, distinct methods are utilized to recognize hazardous scenarios (II) using a SuT to mitigate or prevent them in the subsequent stages of the system's development.

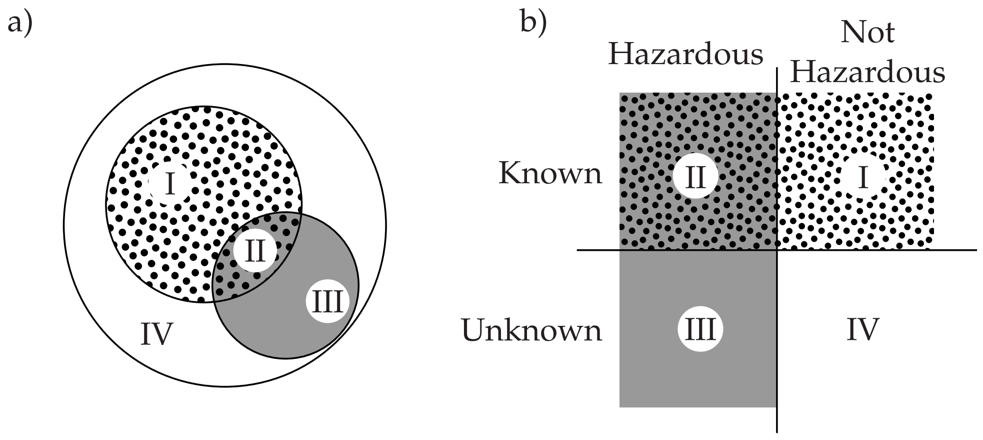


Figure 2.1: Schematic visualization of the different scenario categories according to [61]

2.1.2 Statistical Approaches

The prevailing methodology for non-autonomous vehicles is to assess safety based on kilometers driven (miles driven). Autonomous vehicles are subjected to testing on both public roads and closed tracks in order to accumulate a substantial number of kilometers driven and statistically prove that autonomous vehicles are safer than human drivers [64, 65, 34]. Despite its broad application, the approach has faced criticism for being unfeasible, as already explained in the introduction. This method may apply to a system that is considered final after testing. However, depending on how the safety argumentation is framed, the kilometers driven may need to be provided again with each software update. Moreover, it is not assured that all safety-critical scenarios are accounted for in the distance covered. While these situations accurately represent the environment, they are performed randomly in testing, resulting in limited reproducibility and comparability [34].

A widely used measure is the number of disengagements, which describes the deactivation of autonomous mode by an on-board or remote safety driver. Disengagements are employed to monitor the maturity of autonomous vehicles, for example in the US state of California [66]. The main advantage of disengagement as a metric is its established status, regardless of its lack of an uniform definition. A reduction in the occurrence of disengagement indicates the system's progress in learning, however, it does not guarantee the system's safety [67].

2.1.3 Formal Verification

Formal verification applies mathematical modelling techniques to provide a proof for the safety of autonomous vehicles throughout their ODD [65]. The benefit of utilizing formal verification lies in the provision of safety statements that are guaranteed, eliminating the need for testing. This makes it an economical, scalable solution at a reasonable cost. The Responsibility Sensitive Safety (RSS) [34] and the SFF model [68] both employ formal verification techniques, establishing their prominence in ensuring the safety of autonomous vehicles. The aim is to ensure that autonomous vehicles are not responsible for any accidents and take appropriate precautions to compensate for the mistakes of others [34].

Multiple formal techniques are commonly utilized. They include model checking, reachability analysis, and theorem proving. Model checking, which was first created in the realm of software engineering, aims to check whether software behavior complies with the specified design requirements. When safety requirements are expressed as axioms or lemmas, theorem proving can be a helpful tool for verifying safety by examining worst-case assumptions. Among these formal techniques, reachability analysis holds a unique and significant position, primarily due to its inherent ability to generate safety assertions for dynamic systems. This capability allows reachability analysis to capture the essential characteristics of the dynamic driving task [69]. However, when considering other traffic participants as a black box, a solid formal verification will assume very conservative reachable sets, which may lead to a situation where “safe” planning is not possible. Therefore, it is necessary to make reasonable assumptions in order to drive safely on the road. Yet, this approach may reduce the level of the safety statement and may not be able to prove safety again [34, 69].

2.1.4 Scenario-based Testing

Scenario-based testing is a concept derived from software development. Testers conduct the testing of the SuT by executing a range of discrete runs, which in this case are scenarios, and evaluating the quality of the run on the basis of defined performance indicators. The entire process comprises three main phases: firstly, the generation or extraction of scenarios, secondly, the execution phase and, finally, the evaluation of test cases with risk assessment and safety argumentation in relation to the SuT.

A key requirement for scenario-based testing is the describability and associated parametrizability of the scenarios, including the environment they contain. The scenario environment is arranged using the five-layer model [70, 71], with relevant information divided accordingly. The first level contains information about the road geometry and topology, the second level contains the traffic-related infrastructure (signs, traffic lights, etc.). Level three enhances the elements of the first two levels by adding temporary changes to the road layout or the traffic signals. Level four covers both static and dynamic objects, while the last level describes environmental conditions, such as the weather. The five-level model has a significant role in various related works and forms the foundation for the description of the ODD. The content of this thesis is mainly focused on level four, considering only dynamic objects. The first level also plays an important role in defining the road topology.

To offer a thorough explanation of scenario-based testing, it's essential to begin by clarifying the temporal concepts of a scene and a scenario. Although various definitions of these terms exist, this work primarily adopts the definitions provided by Ulbrich et al. [72] (see Definition 1 and Definition 2). A graphical illustration of a scene (a) and a scenario (b) is shown in Figure 2.2.

Definition 1. A **scene** is a snapshot of the environment, including static and dynamic objects. Each state and intention of the dynamic environment is frozen.

Definition 2. A **scenario** describes the temporal development in a sequence of scenes. Every scenario starts with an initial scene.

The lower part of Figure 2.2 demonstrates the systematic connection between a scene and the chronological sequence of scenes, that is a scenario. The

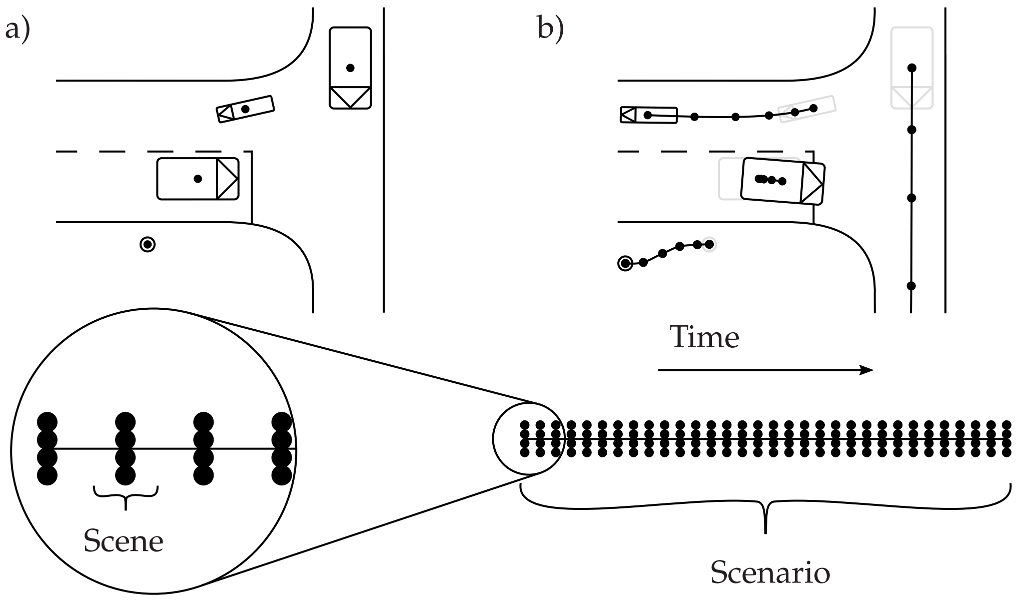


Figure 2.2: Schematic visualization of the relationship between scenario and scene

upper part shows a schematic image of a traffic scene (Figure 2.2a) consisting of two vehicles, a cyclist, a pedestrian and the road. On the right (Figure 2.2b) is a scenario that evolves from the scene on the left. The road remains the same. The states of the dynamic traffic participants change over time, which is represented here by a trajectory of each traffic participant. Due to the challenge of representing a scenario continuously in practice, a process is discretized at fixed points in time. The concrete form of the elaborated scenes and scenarios is described in more detail in Section 3.1.

Scenario-based testing in the context of highly automated driving involves creating a set of hypothetical situations or scenarios that a HAVs might encounter on the road and testing the vehicle's ability to respond appropriately to those scenarios. These scenarios can include a wide range of conditions, such as different weather conditions, road configurations, traffic patterns, and unexpected events such as sudden obstacles or pedestrians crossing the road. Scenario-based testing is intended to ensure that HAVs are capable of handling a variety of real-world situations safely and effectively through targeted testing of specific scenarios. According to the PEGASUS project [43], only a few systematically created test scenarios are sufficient to make a reliable statement about the safety of a HAV. By testing HAVs in a controlled environment, vulnerabilities or limitations in the vehicle's software can be identified and improved to enhance their performance and especially safety. Scenario-based testing methods are an important part of the development and validation process for HAVs. They help to ensure that they can operate safely and reliably under a wide range of conditions and environments.

One of the biggest challenges in scenario-based testing is the correct choice of test scenarios. The more scenes are tested, the better the transferability to reality. However, since potentially infinite scenarios are possible, complete coverage is impossible. Accordingly, a point in time must be found at which testing is sufficient, and a statement can be made about the safety of the HAV. In the work of Sun et al. [73], methods for generating test scenarios are divided into three classes: Coverage-oriented methods where the coverage of the specific ODD is maximized, unsafe-scenario-oriented methods that mainly examine exceptional cases, collisions, and high-risk scenarios to detect failures, and the naturalistic evaluation approaches that focus on deriving safety indicators. In general, these approaches can be categorized as *knowledge-driven*, *data-driven* and *other* scenario generation [47]. Depending on the area of application, these categories can be further subdivided. For example, Birkemeyer et al. differentiate between random and combinatorial scenario generation in the knowledge-based approaches [44].

The PEGASUS project [43] divides the resulted, generated scenarios into three subcategories: functional, logical, and concrete scenarios. Functional scenarios use natural language and present processes abstractly. Logical scenarios have specific parameter ranges. Concrete scenarios, on the other hand, have a fixed set of parameters and represent a fixed sequence of scenes. Concrete scenarios are crucial for the actual implementation of the test process, for example, in a simulation.

Knowledge-driven Scenario Generation

The knowledge-driven approach extracts information from various sources such as existing scenarios, functional descriptions, experts, traffic guidelines and standards. This information is then structured and shared through an ontology or another description model, which provides an abstract representation of the world. In scenario-based testing, this model is employed to organize the components of a scenario and subsequently the ODD, which allows for the generation of scenarios using sampling or combinatorial techniques [65].

As illustrated in Figure 2.3, the essence of knowledge-driven scenario generation is the description model. The level of complexity and detail of e.g. the ontology determines which environments and corresponding scenarios can

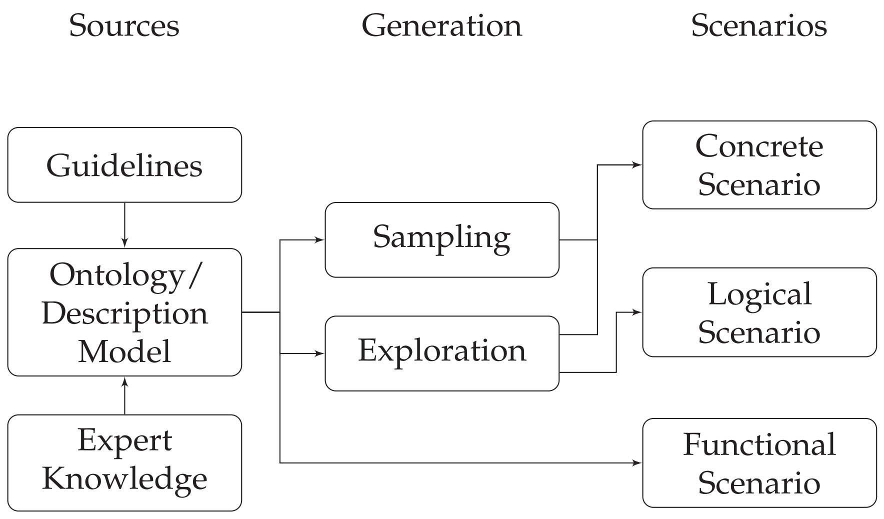


Figure 2.3: Knowledge-driven scenario generation process (Based on [65])

be mapped. Section 3.2 further discusses relevant ontologies in the context of scenario-based assurance of HAVs. Based on the ODD provided by the ontology, new scenarios and constellations of entities can be created using combinatorial methods [74, 75, 44]. Depending on the level of detail of the description model, functional, logical or concrete scenarios are created.

Generating scenarios using knowledge-driven methods have proven to be very convenient to generate and classify. Consequently, a multitude of diverse scenarios can be generated with minimal effort, given certain framework conditions. However, a significant problem is that the domain is limited to the ontology or description model defined by experts or standards. This leads to suboptimal performance when mapping naturalistic scenarios, especially in edge cases [76].

Data-driven Scenario Generation

Like the knowledge-driven approach, the data-driven approach (see Figure 2.4) aims to generate concrete, logical, and functional scenarios. The difference between the two approaches, however, is the source of the information. Data-based approaches rely on specific scenarios that are already known, whether stored in an accident database or obtained from real road traffic or simulations. Real data provides a high degree of realism because it is less bound by the limitations and biases of expert models than knowledge-based methods [57]. This method is well-suited for examining significant or corner cases, as these can be extracted directly from the databases after appropriate evaluation.

One limitation of real data is the significant level of redundancy, since the majority of the data comprises traffic scenarios with high probabilities of occurrence [77]. To enable wider application of the method and to generate functional and logical scenario classes, it is necessary to undergo a grouping and sometimes a classification process. During the classification, specific criteria determined by humans are applied to scenarios. If two concrete scenarios fall within the same value range or meet certain criteria, they are classified into the same group. The boundaries between the classification and grouping procedures are unclear. Nevertheless, a line can be drawn between unsupervised clustering and manually defined methods of classification. Details and a more extensive review of literature concerning scenarios and scenario grouping can be found in Chapter 5. Depending on the density of the clusters, that is, how continuously each parameter distribution is covered, they can serve directly as logical or even functional scenario descriptions. To ensure continuous parameter coverage, additional parameter ranges and distributions can be defined to precisely describe the logical scenarios.

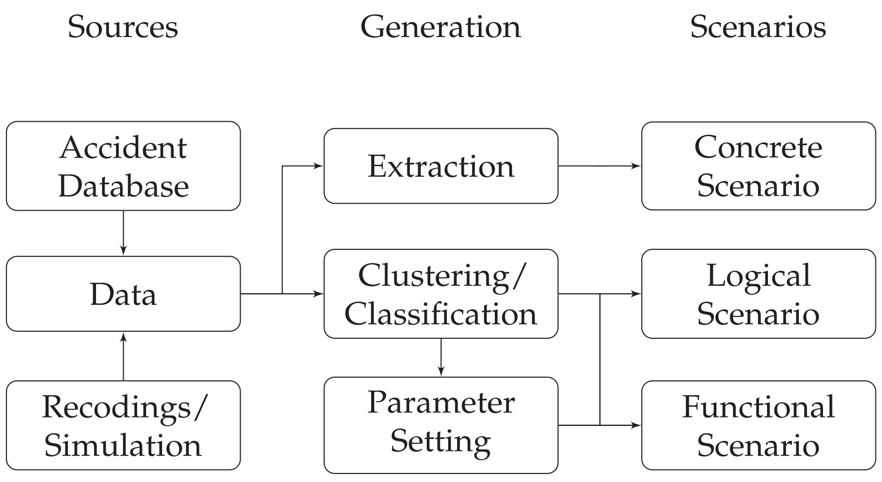


Figure 2.4: Data-driven scenario generation process (Based on [65])

Other Scenario Generation Methods

In addition to data-based and knowledge-based generation methods, there are other approaches that cannot be clearly categorized into one of the two methodologies described above. One approach that has become particularly popular recently is adversarial scenario generation. This involves using a generator to generate a scenario that directly integrates the vehicle (victim). This is why these generation approaches are also called vehicle-in-the-loop [47]. Real data is often used for the initialization of such environments, but the generation is similar to knowledge-based approaches, as the parameters are tailored using various optimization methods [78, 79]. Finding critical scenarios is often formulated as a reinforcement learning problem. This eliminates the need for concrete parameterization of the model, as it learns independently, but via an externally specified policy [47, 80].

2.1.5 Summary

In contrast to the conventional testing of vehicles, the testing of HAVs focuses on the safety of the intended functions in complex driving environments. In view of these changes, a new testing methodology is required. Scenario-based testing has several advantages over the statistical methods: on the one hand, the same scenarios can be tested, but these are encapsulated and can be analyzed separately and, above all, also modified. This means that repetitions can be avoided and important or rare scenarios can be examined in a targeted manner. In terms of the economic aspect, the most effective approach is to conduct a formal verification of a driving function. However, due to underlying assumptions, this is not always feasible and formal verification can only be applied to specific sub-functions.

It is worth noting that it is difficult to draw a clear line between data-driven and knowledge-driven methods in the final implementation. Consequently, **expert knowledge can not be clearly excluded from the data-driven paradigm**. In fact, it is advisable to make use of expert knowledge, especially in the initial phase of scene description, where the collection of relevant information and the reasonable exclusion of irrelevant data can greatly improve the subsequent clustering and classification phase [76]. This ensures a seamless integration of data- and knowledge-based approaches, which is key to the design of an effective test process.

2.2 Graph Neural Networks

The purpose of this chapter is to provide a brief overview of machine learning with graphs. For an insight into the operation of graphs and a very detailed discussion of graph neural networks, see the book by Hamilton [81], which is recommended for a deeper insight.

Graphs are all around us, modeling complex systems with different components and various interactions. Graphs and the information stored in them are described by nodes and especially by edges, which represent connections between a pair of nodes. In the literature, the term vertex is also used in addition to the term node and can therefore be used interchangeably. Formally, a graph G is defined by a set of nodes V and a set of edges E:

$$G = (V, E) \tag{2.1}$$

The edge ($e^{vu} \in E$) points from a node v ($v \in V$) to the node u ($u \in V$). Both nodes and edges can hold attributes, which means they can store a lot of specific information about entities and relationships.

The fundamental concept of machine learning with graphs is largely identical to that of conventional machine learning methods. However, certain distinctive characteristics must be taken into account. Machine learning tasks within the graph domain can be divided into four distinct categories. In node classification, the objective is to predict a type or attributes of a node. In contrast to conventional supervised classification, nodes within a graph are not independent and identically distributed (i.i.d.). Node classification does not consider a single independent data point; rather, it considers a set of connected nodes. The neighborhood of a node has a highly informative value about its properties [81]. In relation prediction class, the aim is to predict missing edges in a set of nodes. A common example of this would be the suggestion of new friends within a social network. The third class of graph-machine-learning tasks is the community detection. In contrast to the node classification and edge prediction tasks, the identification of communities is considered an unsupervised task. The objective is to identify distinct sets of nodes within a large graph that show similar properties, and then assign them to a group. The last category is graph classification. The aim is to predict the properties of an entire graph. Here, not individual components such as nodes or edges are predicted, but the entire graph is taken into account. In most cases, this task is usually trained as a kind of supervised task, where training graphs have a label. One of the most prominent examples is to predict the chemical properties of a molecule represented as a graph. In this dissertation, node and graph classification is primarily applied.

In general, the objective of a Graph Neural Network (GNN) is to generate a representation of a node that is dependent on the structure of the graph and its features. One of the most significant differences between GNNs and conventional Convolutional Neural Networks (CNNs) is that the CNN's input is clearly defined and always follows a specific order. Images serve as a simple example of this, as they are defined by pixels that are arranged within a well-defined grid pattern. This means that virtually every pixel has exactly eight neighboring pixels whose relative position is known. The topology of graphs is frequently described using an adjacency matrix, which itself represents an ordered format. However, the nodes within the matrix can be arranged randomly, thereby describing the same graph. A corresponding model is therefore not permutation invariant. The concept of permutation invariance refers to the ability of a model that employs an adjacency matrix to generate identical outcomes regardless of the rearrangement of rows and columns.

An important concept involved is message passing MP. Here, a node embedding h^i of the node $i \in G$ is updated based on its respective neighborhood nodes $\forall j \in \mathcal{N}(i)$. Message passing propagates a node's information one edge (one hop) at a time. In order to propagate the information further in the graph, the message passing step is repeated several times in a row. An index for the update step $k \in K$ is added to the node embedding h_k^i notation to better assign these "time-dependent" states. h_0^i represents the initial node attributes in the input graph.

$$h_k^i = \gamma_k \left(h_{k-1}^i, \bigoplus_{j \in \mathcal{N}(i)} \left(m_k(h_{k-1}^i, h_{k-1}^j, e_{attr}^{ji}) \right) \right) \tag{2.2}$$

The general formula for the message passing operation is shown in Equation (2.2). The information from a neighbor node h_{k-1}^j $j \in \mathcal{N}(i)$ is processed by the message function m_k. This function can have various parameters, but in most cases the target node state h_{k-1}^i, the neighbor node state h_{k-1}^j, and the edge attributes e_{attr}^{ji} are used. Normally, an ordinary Multilayer Perceptron (MLP) is used, whose weights can be parameterized during training. Recently, also after the success of transformers [82], attention-based message functions are used. The standard transformer layer is identical to a GNN layer, which uses multi-headed attention and which is applied to a fully connected graph [81, 83].

All messages, which are generated by m_k, are then combined using the aggregation function $\bigoplus$. As mentioned above, it is important that this function is permutation invariant with respect to the input. In most cases, the min, max, sum, or mean operator is used. The aggregated messages are then mapped to the new target node state h_k^i using the update function γ_k. Typically, the last target node state h_{k-1}^i and the aggregated messages are processed using a MLP that can be parameterized in training.

Figure 2.5 shows an example graph and a 2-step message-passing operation for the example node 1. Looking at the last layer, it can be seen that node 1

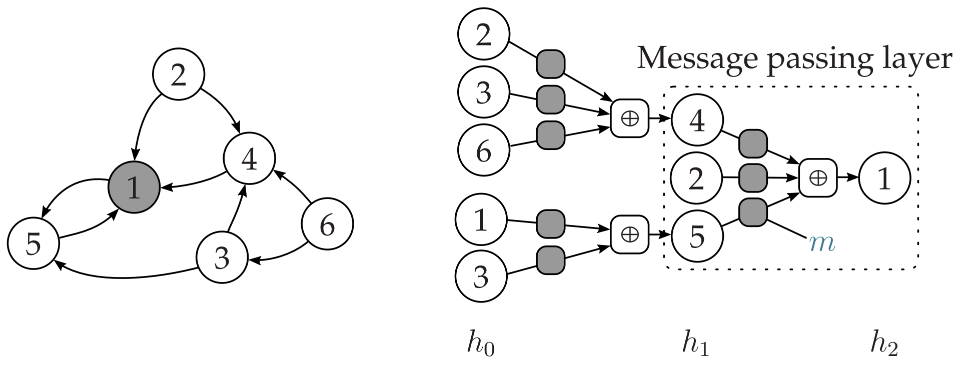


Figure 2.5: Example graph and the corresponding message passing scheme for node 1

is directly dependent on the information of nodes 4, 2, and 5 because they have an edge to node 1. However, each of these nodes' information depends on the message-passing layer that was performed before it. Node 5 in turn depends on node 3 and node 1, and node 4 depends on node 2, node 3, and node 6.

The node embeddings h_k generated by message passing can be used for a node classification task or an edge prediction task. To make a statement about the whole graph, graph pooling (or graph readout) comes into play. Similar to the message aggregation function, a function must be found that converts a set of nodes into an embedding for the full graph. In practice, basic sum or mean operators are used for most graphs. More complex, attention-based methods are sometimes used, especially for larger graphs [81].

3 Scene Representation

Arguably, the key element in a data-based processing chain is the selection of an appropriate environment representation and the associated processing of the output data. This representation determines the overall information that can be represented and the manner in which it can be accessed. In essence, the representation should create a model of the world that, ideally, contains all the information relevant to the subsequent task. In the context of highly automated driving, a variety of information sources, such as sensors, maps or other communication interfaces, are often merged. This representation serves as an abstract interface for the subsequent driving tasks. The 5-layer model [70] provides a reasonable categorization of the various environmental factors that can be relevant for highly automated driving.

In road traffic, processes and entities follow certain rules. The more structured and repetitive the environment, the more the representation model can abstract it without losing important information. Selecting an appropriate descriptive model relies heavily on the intended downstream task. This is because the focus of the abstraction model should be tailored to the objectives being pursued. The selection of an appropriate description model is based on the assumption that it contains important information for the selection of relevant scenarios for the testing of driving functions.

This section describes the dataset used in this thesis, and discusses various methods for creating descriptive models. Finally, a detailed explanation of the description model created specifically for this thesis is provided, which tries to answer Research goal 2.

Parts of this chapter have been published previously in the following peer-reviewed publications [8, 1, 2], and some of the content has been adopted verbatim.

3.1 Input Data

Real data serves as the foundation for all subsequent investigations presented in this work. Numerous publicly available traffic datasets exist [84, 85, 86, 87, 88, 89], each with distinct characteristics and information. As a result, certain datasets are suitable for specific applications while others are not. The following section briefly outlines the necessary data requirements and information notation.

This work examines the behavior and social interactions of traffic participants in the form of their trajectories and scene constellations. The prerequisite for the data to be further processed is that for each desired point in time, all traffic participants I can be assigned to a discrete state. The state $\mathcal{X}$ of the traffic participant i ($i \in I$) is defined as follows:

$$\mathcal{X}^i = \{x^i, y^i, \psi^i, \dot{x}^i, \dot{y}^i, \ddot{x}^i, \ddot{y}^i, w^i, l^i, \kappa^i_{obj}\} \tag{3.1}$$

The traffic participants are considered exclusively in a planar $\mathbb{R}^2$ coordinate system. Here, x^i and y^i describe the geometric center of the traffic participant in a local metric coordinate system, whose origin is usually defined by a geodetic reference point. ψ^i specifies the orientation (yaw angle) of the vehicle. $\dot{x}^i$, $\dot{y}^i$ and $\ddot{x}^i$, $\ddot{y}^i$ specify the traffic participant's two-dimensional velocity and acceleration, respectively of the traffic participant, in the direction of the respective axis. The dimensions of the respective object box is described by the width w^i and the length l^i. In addition to the geometric information, an object is described by its classification κ^i_{obj} (*Car*, *Pedestrian*, *Truck*, ...).

The states $\mathcal{X}$ of all traffic participants at a given time step are then combined to a scene and described by the scene state $S_{\mathbb{R}^2}$(t):

$$S_{\mathbb{R}^2}(t) = \mathcal{X}^i(t) | i \in I\}. \tag{3.2}$$

In general, any dataset or synthetic data that meets the above requirements and provides an HD map can be used for the analyses shown below. Object list motion datasets are particularly well suited for trajectory analysis. Prominent examples are the InD [86] and HighD [87] datasets, the NuPlan dataset [89], and the INTERACTION dataset [85]. Furthermore, the TAF-BW dataset [1], which focuses on trajectory data in combination with traffic light control, was generated in the course of this work and forms an additional dataset.

The INTERACTION dataset is used for all further investigations. This dataset captures the behavior of vehicles on roads in different countries, with a focus on intersections and roundabouts. Compared to the TAF-BW dataset, the INTERACTION dataset contains far more data and more different types of intersections, which have also been recorded in other countries with different traffic regulations, making it significantly more valuable. Figure 3.1 shows exemplary traffic scenes on the various road maps contained in the dataset. The utilization of a drone for recording helps to provide consistent observation of identical locations in each sequence. This is an advantage over

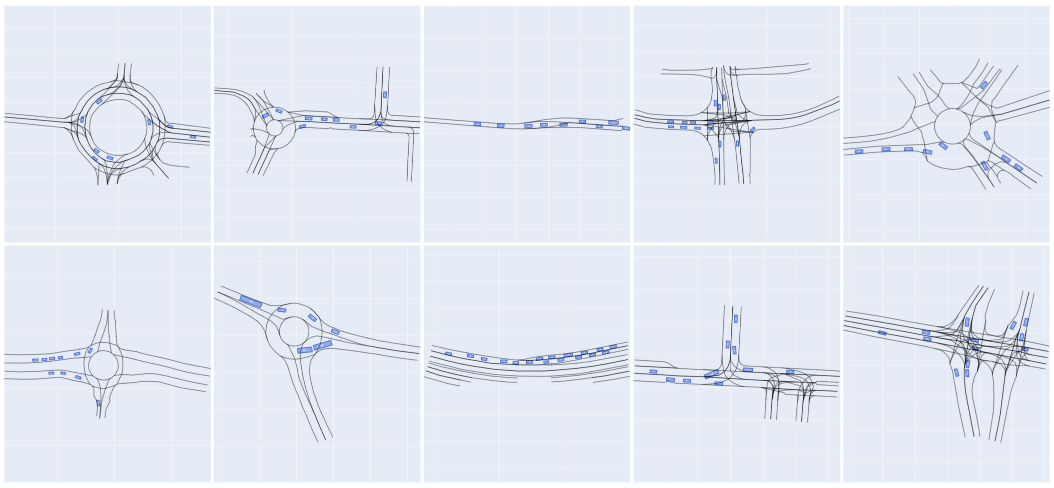


Figure 3.1: Road maps of the INTERACTION dataset. The black lines indicate lane boundaries or road markings, while the blue rectangles represent the pose of traffic participants.

datasets that have been recorded from the perspective of a vehicle (e.g. the NuScenes dataset [89]), as it makes it easier to compare the behavior of traffic participants. This is based on the assumption that the environment, apart from the traffic participants, remains identical, and changes in behavior can be largely reduced to other dynamic traffic participants. In addition to object tracks, the dataset also comprises corresponding HD maps of the roads. Each object in the dataset is defined by its pose in Cartesian space, classification, and velocity, sampled at a rate of 10 Hz.

3.2 State of the Art

Environment representations can be classified according to their level of abstraction and how the various elements are related to each other. Bringing together a vast range of different sources of information (e.g. different sensors) requires the development of a robust and comprehensive environmental model. This model serves as a distilled representation of the actual world and provides the basic framework for all subsequent tasks. The greater the degree of abstraction, the more the model shows increased generality in terms of its ability to represent different environmental contexts. However, it should be noted that models with less abstraction hold more information. Overly complex models can pose difficulties in terms of interpretation in subsequent processes [53].

3.2.1 Spatial Representation

The most abstract and therefore the most compact representation of the environment are feature maps or object lists. These use single points in space for objects and their associated features to represent the environment. Motion datasets are often presented in this simplified format to minimize storage requirements, focus the information they contain on a specific task and preserve the anonymity of the recorded traffic participants (see Section 3.1).

Moving towards a more granular level of abstraction, grid maps are a popular representation of the environment. This approach divides the environment into a finite number of cells, which are often uniformly distributed across 2D space. This variant can occur in a simple binary form; such an occupancy grid shows whether the cell is occupied and whether, for example, an object is located there. Taking this concept further, cells can be classified to indicate the type of object within the area. This results in the commonly used birds-eye view, which represents the environment as an image where each pixel corresponds to a cell. This representation has been particularly popular in recent years for birds-eye-based trajectory predictors [90, 91, 92]. A detailed discussion of trajectory predictors and the representations used are explained in more detail in Section 6.1.

The most detailed level of abstraction is where the sensor data is directly visualized. This could include point clouds from Light Detection and Ranging (LIDAR) sensors or camera images, for instance. Several intermediate levels that cater to specific use cases exist between the mentioned individual abstraction levels. For a comprehensive analysis of various spatial environment representations, refer to the survey of Schreier [53].

Spatial representations have become increasingly popular in recent years, especially in the context of data-driven approaches. This may be explained by the accessibility and availability of the underlying data [89, 88, 85, 86, 87], but also by the convenience of processing such data. In particular, images that can represent a grid map, for example, can be processed surprisingly efficiently and with little effort using CNNs.

A decisive disadvantage of spatial representations is the neglect of explicit dependencies and relations between objects. In reality, objects follow certain rules. For instance, a model can predict a trajectory that follows the course of the road utilizing a top-down grid map where the road boundaries and the road markings are depicted. However, if the relationships and dependencies between entities become more complex and cannot necessarily be derived from the object's position, they can no longer be captured using spatial representations. This disadvantage is addressed through the use of topological representations.

3.2.2 Topological Representation

In contrast to the exclusively spatial approach, parts of the environment can be described using topological models. In this case, the environment is defined as a set of distinct objects. It is possible to explicitly incorporate interactions into the modelling process. However, it is not feasible to represent all minor details, and therefore the interactions must be as straightforward as possible.

Ontologies are suitable for such an explicit description of objects and, in particular, their interactions. Within computer science, an ontology structurally models a domain and can be adopted by an application as a knowledge repository [71]. As per Studer et al. [93], *"an ontology is a formal, explicit specification of a shared conceptualization"*. The key benefits of ontologies result from the conceptualization described by Studer et al. [93]. Firstly, ontologies have the characteristics of being shared and explicit, which allows them to encapsulate common knowledge and provide a vocabulary without ambiguity [94]. Secondly, ontologies are characterized by their formal language specification, which enables computational processing and promotes human interoperability and reusability [95]. Thirdly, ontologies rank concepts and relations in a hierarchical taxonomy, creating a structural framework that facilitates knowledge deduction through semantic connections and set regulations [96]. In terms of architecture, an ontology can be divided into terminological boxes and assertion boxes. The terminological boxes contain the general background knowledge consisting of the concepts, relations, attributes, axioms and rules. The assertion boxes contain instances or specific objects and relations between them. For further information, refer to the work of Studer et al. [93].

In the field of autonomous driving, ontologies serve two primary purposes. Firstly, they are utilized to generate scenarios for knowledge-driven scenario-based testing [75, 74, 71, 97, 98]. Secondly, they are applied to evaluate the perceived environment of the automated vehicle, assess the situation, and determine the next course of action based on these results [99, 100, 101, 102, 103]. The core idea remains the same, regardless of any variations in the original concept: to describe the environment and its significance in executing the highly automated driving task.

Klück et al. [75] and Wotawa et al. [74] leverage ontologies to define influence factors and test parameters (like road parameters or vehicle dynamics), transforming the ontology into an input model for a combinatorial algorithm that generates test suites of abstract scenarios. These scenarios encompass road infrastructure and ego-vehicle position and speed. Similarly, Bagschick et al. [71] construct an ontology for scenario modelling, encompassing road, traffic infrastructure, temporary road manipulations, static and dynamic objects, and the environment layer, where the description of the environment is based on Schuldt's five-layer model [70]. Chen and Kloul [104] employ a three-layered methodology for the generation of test scenarios. This methodology incorporates a highway, weather, and vehicle ontology for the modelling of static and mobile scenes. Additionally, it incorporates first-order logic rules for the description of interactions and a generation layer for adding scenes with actions and events.

The simulation scenario generation framework of Medrano-Berumen and Akbas [98] use a matrix where each row represents a road piece or actor in a semantic string. The string defines a road piece with parameters including type, length, lanes, speed limit, and intersection pattern. For the description of actors, parameters, such as actor type, path type, moving speed, start location and offset are included.

Elgharbawy et al. [105] take a data-driven approach, using various data sources to identify driving situations and cluster data into groups. They utilize data regression to generate scenarios based on characteristic signals. Herrmann et al. [106] provide another example of how ontologies can be applied to a data-driven approach. They use ontologies to describe images of traffic situations, create training and test datasets, and annotate data. Their ontology considers various concepts, including lighting conditions, pedestrian clothing, buildings, and road conditions.

In the context of scenario-based testing, Bogdoll et al. [107] propose a master ontology for modeling diverse corner cases for autonomous vehicles. This ontology can be converted into the OpenSCENARIO [108] format for simulation testing, enabling the evaluation of challenging scenarios not encountered during training data collection.

Several studies have explored the application of ontologies for enhancing decision-making in autonomous vehicles. Ulbrich et al. [99] developed a graph-based scene representation similar to the W3C Web Ontology Language [109], that consolidates information from environment perception modules, maps, and Vehicle-To-Everything (V2X) data, enabling context modelling and informed decision-making. They include details about the ego-vehicle, other objects, infrastructure, and knowledge of the situation, including planned routes and actions, along with constraints representing traffic rules. Hülsen et al. [100] demonstrate the significance of reasoning on sensor data in decision-making, introducing relations such as *hasToYield* and *hasRightOfWay*. They focus on improving decision-making at intersections, emphasizing the need to predict future maneuvers. This approach emphasizes the importance of high-level semantic traffic models and lane relations. Buechel et al. [101] also adopt an ontology-based approach for decision-making but with a broader scope, encompassing generic traffic rules and national regulations. Their ontology facilitates cross-country applicability by modelling rules in a generic manner whenever possible. Contrarily, when generic modelling is not feasible, they represent traffic rules as a class instance. Regele et al. [102] align with the focus of Hülsen et al. on traffic coordination at intersections, but distinctively address the challenge of choosing the right abstraction level for the model. They propose a two-tier model, distinguishing between a low-level world model for trajectory planning and a high-level model for traffic coordination, prioritizing semantic meaning for quick decision-making.

Fang et al. [103] recommend a probabilistic approach for forecasting long-term behavior of nearby traffic participants, with an ontology representing encountered situations and behavioral intention series, which let them model future interaction of traffic participants. In contrast, Huang et al. [110] take a deterministic approach for decision-making, using ontology to assess scenarios around the ego-vehicle. Like Huang et al. Kohlhaas et al. [111] utilize eight regions around the considered vehicle that describe its environment and its spatial interaction with other vehicles in a semantic way. This state space includes possible transitions and their associated probabilities and validity, enabling high-level maneuver planning. Petrich et al. [112] present a tensor-based approach that considers relationships between entities in combination with the environment to model high-level context information. This approach generalizes over a variety of scenes (referred to as situations) by using a unified description.

Zhao et al. [113] transform sensor data into a Resource Description Framework (RDF) and store it in an ontology. A query engine retrieves information about the environment, position, and direction of the vehicle from the ontology. The ontology consists of three sub-ontologies that can represent the topology of the road, the type of traffic participants and their behavior.

Many of the ontologies shown above can represent the environment very accurately, and in particular the ontologies developed for scenario-based testing can also represent the processes and intentions of traffic participants well. The consensus of the ontologies, which is also particularly pronounced in the decision-making ontologies, is the importance of the road topography. Furthermore, in almost every ontology, the position and the dynamic state are defined to some extent. Table 3.1 displays a summary of the most prominent categories considered in the ontologies reviewed. While one advantage of ontologies is the ability to precisely define the relationship between entities, information beyond the spatial content can typically be represented in the application through manual labelling or only by ground truth information, for example in a simulation. This makes it challenging to process the information in an automated, data-driven manner.

	Road Geometry	Road Condition	Road Markings	Road Type	rel. Position	State of TP	Intentions/ Actions	Traffic Rules	Static Objects	Signs	Environment
Bagschik [71]	✓	✓	✓		✓	p	✓	✓	✓		✓
Klück [75]	✓	✓	✓			pv					
Wotawa [74]						pva	Driver Type				
Chen [97]	✓		✓	✓	✓	pv	✓		✓	✓	✓
Medrano-Berumen [98]	✓			✓		pv		✓			
Schuldt [70]	✓		✓	✓	✓	v			✓		
Elgharbawy [105]			✓			p			✓		
Ulbrich [72]	✓			✓		p	✓	✓		✓	
Hülsen [100]	✓			✓		p	✓	✓		✓	
Büchel [101]	✓		✓	✓	✓	pv	✓	✓	✓	✓	✓
Regele [102]	✓			✓		pva		✓		✓	
Bogdoll [107]					✓	pva	✓				✓
Huang [110]	✓		✓	✓	✓	pv	✓		✓	✓	
Zhao [113]	✓	✓	✓	✓		pv	✓			✓	
Fang [103]	✓			✓	✓	pv	✓			✓	
Kohlhaas [111]	✓				✓	pv					
Herrmann [106]		✓				p			✓		✓
Westhofen [114]	✓		✓	✓	✓	pva	✓	✓	✓	✓	✓

Table 3.1: Representability of different categorizations by the investigated ontologies. The state of traffic participants (State of TP) is described by the position p, the velocity v and their acceleration a (Adapted from [2]).

Ontologies often tend to annotate things very precisely, which inflates the model and can make it difficult to generalize. For a detailed comparison of ontologies in the context of safeguarding autonomous driving, please refer to the paper by Zipfl et al. [2].

3.2.3 Graph-based Representation

In recent years, there has been a trend to represent the environment in the form of graphs, especially for predicting the behavior of other traffic participants (see Section 6.1). Ontologies, constructed with RDF, can be viewed as a type of heterogeneous graphs. This thesis makes a distinction between knowledge graphs and plain networks, which will be referred to as graphs in this work. Graph-based description models, on the other hand, tend to be more concise and integrate both numerical and semantic information.

In the work of Zhou et al. [115] a fully connected graph is used to describe a scene for pedestrian prediction. Each node within the graph corresponds to a pedestrian. A second graph is added for each pedestrian and its past trajectory to capture the temporal dependency. The weighting of edges is subsequently determined using a learned process that is dependent on the scene. Yu et al. [116] employ a similar approach, but only span undirected edges between pedestrians that fall below a certain distance threshold. Other approaches [117, 118, 119] for the prediction of movements differ only slightly in their representation of the environment. The graphs are usually either fully connected or have edges only between nodes that are spatially close. Gao et al. [120] propose an approach called VectorNet that employs an environment representation of both map and traffic participant features to learn ego trajectories. Both map elements and traffic participants are represented by nodes. All nodes collect information as a fully connected graph within a certain radius around the root node.

Diehl et al. [121] develop an early approach to link vehicles in a highway scene using edges. The nearest eight vehicles (similar to the work of Kohlhaas et al. [111]) are chosen as the condition for an edge. Ma et al. [122] introduce an approach that utilizes an image based machine-learned pre-processing step to imitate human prior scene understanding, including understanding of pairwise relationships between agents and inferred pairwise contextual information. Edges have different attributes, such as the Euclidean distance or the distance in Frenet coordinates between traffic participants. In contrast to previous graph representations, only traffic participants who influence each other's behavior are connected by edges.

A thorough comparison of various descriptive models concerning the emphasis on social relations within the road traffic context is developed in the survey of Wang et al. [123]

3.2.4 Summary

The weighting of pieces of information about a scene is dependent upon the type of scene representation. This is either because the information can not be represented in an effective manner or because it is unimportant for the respective tasks. Current scene descriptions often focus on spatial information and neglect interactions between entities, as shown in the above excerpt from the state of the art. However, to describe the traffic scene accurately, it is essential to describe the social interactions (see Definition 3) between individual traffic participants [123, 124, 125]. These interactions determine the future course of the scenario and the behavior of different actors.

For this reason, ontology-based description models take into account relationships between individual traffic participants and static objects. Specific

> **Definition 3.** The **Social interaction** that occur within the context of road traffic are characterized by a dynamic sequence of actions, which are shaped by the actions and reactions of individuals. This is achieved through an information exchange process between two or more agents, with the aim of maximizing benefits and minimizing costs *cf.* Wilde et al. [123].

interactions and relations are considered, resulting in a large ontology that can be challenging to use when describing a traffic scene at high detail level.

The graph-based approaches combine the advantages of spatial approaches by deriving and calculating automatically generated features based on spatial representations, while retaining the explicit relationships and object features between nodes through defined edges. It is important to note that the level of detail is not as precise as with topological representations. The lightweight representation offers advantages for both data storage and processing performance [121]. Furthermore, a Frenet space representation is particularly generalizable, allowing for different traffic scenes regardless of the road geometry and even the constellation of traffic participants [122].

Traditional approaches to describe traffic scenes make it challenging to compare scenarios involving varying road geometries, locations or traffic participants on a scalable basis. It is necessary to develop a model that can describe a broad spectrum of traffic scenes. This enables similar scenes to be grouped and classified, while still containing enough information to understand the behavior of traffic participants. Recent graph-based approaches create edges between entities based on spatial proximity, which is certainly an important factor. However, vehicles that are not in the immediate vicinity also have a relevant influence on the behavior and thus on the whole situation. In order to map such relations without using a fully connected graph, it is necessary to identify an appropriate approach to reflect this. It is also crucial to minimize the number of entity and relation types. Based on the most important and most represented categories in ontologies for scene description from Table 3.1 and the important points elaborated above, a lightweight, generalized scene description must be found. To achieve this, the Semantic Scene Graph (SSG) model is introduced and applied.

3.3 Semantic Scene Graph

The Semantic Scene Graph model allows the description of traffic scenes according to the road topology and facilitates the comparison of scenes independent of their location. Traffic participants are described as nodes.

The relations between traffic participants are represented by semantically categorized (longitudinal, lateral, intersecting) edges. This results in a scene description that neglects the road geometry and places traffic participants within the road in a relationship to each other. Attributes on the edges further specify the relationship, while attributes on the nodes describe the respective traffic participants. This abstract description is designed to facilitate machine readability, making it practical to apply machine learning methods to traffic scenes. GNNs can examine and process SSGs. Improved understanding of other traffic participants could improve the decision-making process of a driving mechanism (see Chapter 6).

3.3.1 Traffic Data, Road Data, and Additional Information

The basic components of the automatic generation of the SSG for a traffic scene are the object list and the corresponding road information, which are described in detail in Section 3.1. It is crucial to include information about the road topology, such as which lanes are connected and how they are connected.

The road map is described by a directed graph G_{road} defined by (V_{road}, E_{road}). Each elementary road segment (lane) is described by a node $v_{road} \in V_{road}$. Two different road segments (v^a_{road}, v^b_{road}) can be connected by edges $e^{ab}_{road} = (v^a_{road}, v^b_{road}, \kappa^{ab}_{road}) \in E_{road}$. The respective edge attributes κ^{ab}_{road} indicate how two road segments are connected to each other. Two road segments can be located next to or behind each other in the direction of travel. In the latter case, the directed graph can clearly identify the predecessor and the successor. In addition, two lanes can cross or overlap each other, for instance at a road junction. Figure 3.2 shows an exemplary road network and the resulting road graph, where each elementary road segment is represented by a node. Each road segment in Figure 3.2a is shown as a node in Figure 3.2b. The segment A lies topologically next to segment E and proceeds in the same direction, therefore the corresponding nodes have an *adjacent* relationship. The elementary road segments A, B, G form a roadway. Thus, the corresponding nodes (A, B, G) in Figure 3.2b are connected with directed *consecutive* edges. If two elementary road segments overlap, a bidirectional *overlapping* edge is found in the road graph. This can occur when two roads intersect (as seen in C and B) or when two lanes merge into a single lane (as seen in F and C).

In addition to the road topology, traffic signs and the resulting traffic rules are of great importance. Information regarding the traffic rule situation is either deducted from the explicit or implicit road data (e.g. traffic signs,

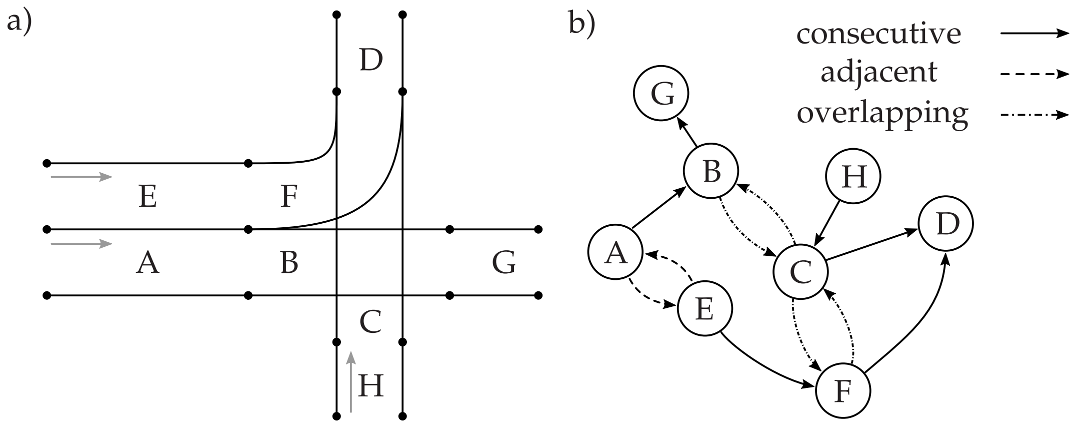


Figure 3.2: A geometric depiction of elementary road segments (A, B, C, D, E, F) defined by the lane boundaries (a). The right illustration shows he resulting road graph; possible relations (consecutive, adjacent, overlapping) are defined by edges between the road segments' nodes (b). (Adapted from [8])

right of way, ...) or parsed from additional information sources (e.g. time-dependent signal phases for the traffic lights). This additional information can be appended with each respective road segment, and subsequently be stored as node attributes in the corresponding graph. However, it is important to note that the dataset used in this work (see Section 3.1) does not include traffic signs, explicit right-of-way rules, or traffic light information. Therefore, the SSG only takes into account the pose of traffic participants and their corresponding positions on the road.

3.3.2 Spatial Abstraction

In order to comprehend a traffic scene, it is crucial to consider the relative distances and velocities of the traffic participants, as they have a significant impact on each other's behavior. Absolute poses in the Cartesian space are of lesser importance. In essence, the most crucial aspect of traffic dynamics is the manner in which traffic participants interact with one another in the context of the road, rather than the specific location where this interaction occurs. In the proposed approach, the traffic participants described by objects in Cartesian space are projected onto the Frenet space [126]. The road segments' centerlines represent the curves. Consecutive road segments of a path in the road network are combined and represented by one curve.

The projection is the first phase in the process of combining the road topology and the object lists. Each traffic participant of a traffic scene in the Cartesian space $S_{\mathbb{R}^2}$ is individually assigned to one (or more) elementary road segment(s) a, depending on its pose in space. This assignment is defined by a projection identity m_a^i. To ensure accuracy, it is important to consider all potentially relevant projection identities m^i in near proximity, in the case of ambiguous poses of a traffic participant i, as some spatial information may be lost during projection (compare Figure 3.3). Depending on the position and orientation relative to the road segment a, a probability $P_{proj}(m_a^i)$ is estimated how well the actual pose of the traffic participant i matches the road segment a. The maximum probability is achieved when the traffic participant is placed in the lateral center of the road segment and perfectly aligned with its orientation. To determine the $P_{proj}(m_a^i)$, the Gaussian function is used:

$$f_d(d_n) = \exp\left(-\frac{d_n^2}{2\sigma_d^2}\right) \tag{3.3}$$

$$f_p(\Phi) = \exp\left(-\frac{(\cos(\Phi)-1)^2}{2\sigma_p^2}\right) \tag{3.4}$$

$$P_{proj}(m_a^i) = f_d(d_n^{ia}) \cdot f_p(\Phi^{ia}). \tag{3.5}$$

d_n^{ia} and Φ^{ia} describe the Euclidean distance of the traffic participant i to the centerline of the road segment a and the angular deviation between its centerline and the orientation of the traffic participant. The standard deviations σ_d and σ_p allow the customization of the specificity to more distant or more rotated road elements. To restrict the number of road segments taken into account for a traffic participant, the search range for matching identities can be limited by distance. Additionally, threshold values can be set for the functions f_d and f_p. The probabilities $P_{proj}(m^i)$ of all m^i of a traffic participant i are then normalized to 1.

In the shown example in Figure 3.3a, vehicle i is displayed on top of three road segments (K, L, M). The distance $d_n^{i\mathrm{K}}$ between i and K, as well as the angular deviation $\Phi^{i\mathrm{K}}$ between the centerline of the lane and the orientation, are displayed. As shown in Figure 3.3b, each projection identity m_{K}^i, m_{L}^i, m_{M}^i is aligned with its respective track. The opacity indicates how high the matching probability is. Here, the vehicle i is spatially farthest from the centerline of M compared to the other tracks. In addition, the angular deviation is also largest here. Consequently, the matching probability is lowest in this case. All vehicles, such as cars, bicycles, trucks and motorcycles, are mapped to the tracks in the same way. Pedestrians, on the other hand, are projected onto nearby lanes regardless of their orientation. The motivation for this is that pedestrians rarely move, similar to vehicles, on and along the road, but cross it or walk on a nearby sidewalk. Therefore, they are mapped

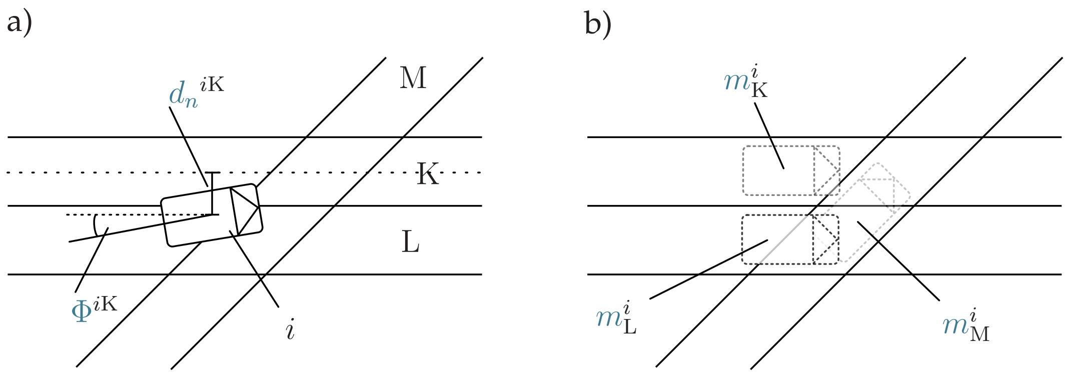


Figure 3.3: Top-down view of vehicle i (and its state $\mathcal{X}^i$) on three road segments K, L, M (a). Three resulting projection identities m_K^i, m_L^i, m_M^i form the projected state $\hat{\mathcal{X}}^i$ of i (b). (Adapted from [8])

to the nearest road segments to avoid neglecting them in the abstraction of the environment.

The result of this projection is that each traffic participant i is linked to a set of elementary road segments by its projection identities m^i defined by $\hat{\mathcal{X}}^i$.

3.3.3 Creation of the Semantic Scene Graph

In order to represent significant relationships between individual traffic participants, their position in relation to other traffic participants I is considered with respect to the road graph G_{road}. The principles of these relation types follow ideas of human behavior patterns. Semantic relationships, such as being in the same lane, adjacent lane, parallel lane, or intersecting lane, are the decisive factors. This information is already implicitly contained in the road graph, in combination with the matched traffic participants. However, in this representation, it is decided to specify the relations explicitly. Possible routes along the road network are taken into consideration to determine the relations between traffic participants' projected states $\hat{\mathcal{X}}$.

Figure 3.4a shows an exemplary projected traffic scene with five vehicles amd their projected states. Every vehicle, except vehicle 1 (m_E^1, m_A^1), has only one projection identity for the sake of simplicity. A depiction of the resulting Semantic Scene Graph $G_{scene} = (V_{scene}, E_{scene})$ is shown in Figure 3.4b. In the following part, it is discussed how to specify different edge classifications κ_{rel}. Furthermore, all attributes of an edge $e_{scene} = (i, j, e_{attr}^{m_a^i m_b^j})$ are described in detail.

$$e_{attr}^{m_a^i m_b^j} = \{\kappa_{rel}, P_{rel}, d_F, d_{ip}, a, d_n^{ia}, \Phi^{ia}, b, d_n^{jb}, \Phi^{jb}\} \tag{3.6}$$

Each traffic participant i is represented as a node $v_{scene}^i \in V_{scene}$. Each v_{scene} stores only the classification type of the object and its absolute velocity (see Equation (3.7)).

$$v_{scene}^i = \{\kappa_{obj}, \left|\dot{x}^i, \dot{y}^i\right|\} \tag{3.7}$$

To determine the edges between nodes v_{scene}, the projected state $\hat{\mathcal{X}}$ of each traffic participant is used, thus comparing all projected identities m (see Section 3.4). Given that multiple projection identities m^i may be assigned to a single traffic participant i (and its node v_{scene}^i), parallel edges between two nodes within the graph may be formed (see node 1 in Figure 3.4b). This can even lead to two parallel edges with the same type κ_{rel}, for example of the merging of the lanes between m_E^1 and m_H^3 and between m_A^1 and m_H^3 where the lanes intersect. In order to ensure the unambiguity of the stored information for several projection identities of a traffic participant, the information is not stored in the nodes but in the edges, as shown in Equation (3.6) and Equation (3.7). Within the SSG, edges are spanned between the nodes, i.e. the traffic participants. However, since the individual projection identities are decisive for this, the notation of an edge between projection identities is used in the further course of the work for the sake of simplicity.

The probability of an edge P_{rel} between two projection identities (m_a^i and m_b^j) is defined by Equation (3.8). This value enables the sorting of multiple parallel edges and enhances readability. This value is relevant when combining several parallel edges into a single edge, as illustrated in Equation (3.15).

$$P_{rel}(m_a^i, m_b^j) = P_{proj}(m_a^i) \cdot P_{proj}(m_b^j) \tag{3.8}$$

Longitudinal Relation

Let p_{road}^{ab} be a path in the road graph G_{road} between two road segments v_{road}^a and v_{road}^b (represented as nodes in the graph) on which there are two projection identities m_a^i, m_b^j. If all edges of p_{road}^{ab} carry the attribute *consecutive*, the two traffic participants i, j have a *longitudinal* relation with each other. As a result, an edge e_{scene}^{ij} between v_{scene}^i and v_{scene}^j can be spanned with the attribute semantic classification κ_{rel} = *longitudinal*. These can be seen, for example, in Figure 3.4 between m_E^2 and m_D^4 (nodes 2 and 4). This means that the two vehicles travel in the same lane, one after the other.

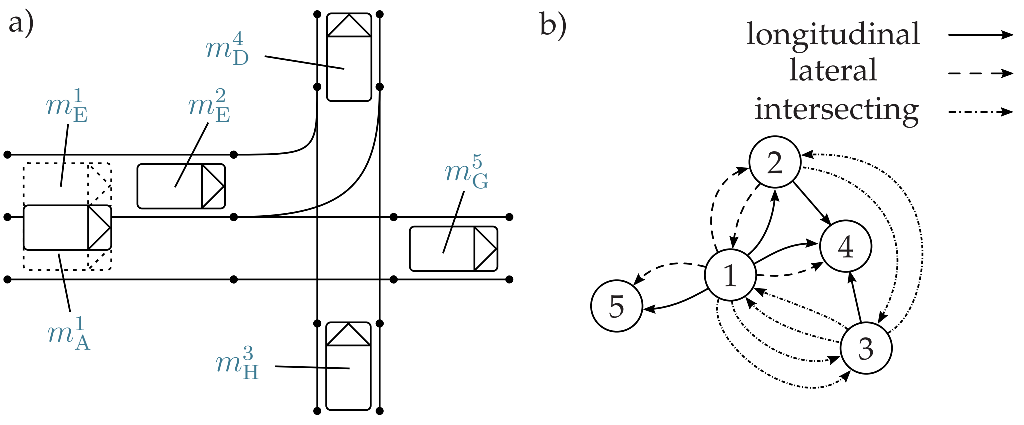


Figure 3.4: The same road network as in Figure 3.2 with six projection identities m of five vehicles (a). The resulting scene graph, where each traffic participant is represented by a node and the relations between its projection identities as edges (b). (Adapted from [8])

Lateral Relation

Vehicles traveling in adjacent lanes are connected in the SSG with a *lateral* relationship. That means as soon as a path p_{road}^{ab} between v_{road}^{a} and v_{road}^{b} exists in G_{road}, where each edge in p_{road}^{ab} has the attribute *consecutive* and exactly one edge *adjacent*, those traffic participants in the scene graph carry the attribute κ_{rel} = *lateral* (see in Figure 3.4 node 1 and node 4). Lateral relationships are always bidirectional. This means, if in a directed graph there exists an edge e_{scene}^{ij} with a *lateral* attribute, there also exists an edge e_{scene}^{ji}.

Intersecting Relation

Intersecting traffic participants i, j (κ_{rel} = *intersecting*) traveling in lanes that will overlap or merge. That means there exists a path $p_{road}^{ab} \in G_{road}$ where each edge of p_{road}^{ab} carries either the attributes *consecutive* or *adjacent* and exactly one edge carries the attribute *overlapping*. Note that once an edge with the attribute *overlapping* is in p_{road}^{ab}, all subsequent edges must be reversed. This is shown in Figure 3.4 at node 1 and node 3. In the corresponding road network, parts of the road segments C and B lie geometrically on top of each other (see Figure 3.2). Looking at the path $p_{road}^{\mathrm{AH}} = \{\mathrm{A}, \mathrm{B}, \mathrm{C}, \mathrm{H}\}$ in the corresponding road graph, it can be noted that paths for *intersecting* relations contain reverse-edges (e.g. e_{road}^{HC}).

In practical use, it sometimes makes sense to limit the length of the path $|p_{road}|$. The number of edges as well as the corresponding length of the road can serve as a limit.

In addition to the semantic classification of the relations κ_{rel}, important parameters are included as attributes in the edges. If the relation is not an intersecting one, the distance (in Frenet coordinates) d_F between both traffic participants' projection identities (m^i, m^j) is stored. Otherwise, the distance to the intersection point d_{ip} is calculated. It should be noted that d_F and d_{ip} can also be negative, since they are measured in the direction of travel. The edge attributes also store information about the projection identities m^i_a and m^j_b. Furthermore, the distance d_n between the projected position and the original position is calculated. Additionally, the angular deviation Φ between the traffic participant's orientation and projection's orientation are stored in e_{scene}. Thus, for each edge, the distances of the projection identities (to the original traffic participant's pose) and additionally the affiliation to each road segment a, b are provided by Equation (3.6).

As stated before, the resulting SSG combines all projection identities of a traffic participant into one node. Information regarding the projection identities can be stored within the corresponding edges. If the projection identities are to be chosen as separate nodes, the scene graph's size would grow drastically, which would reduce the interpretability and inference of the graph. Furthermore, this maintains the affiliation of the projection identities to the actual traffic participant.

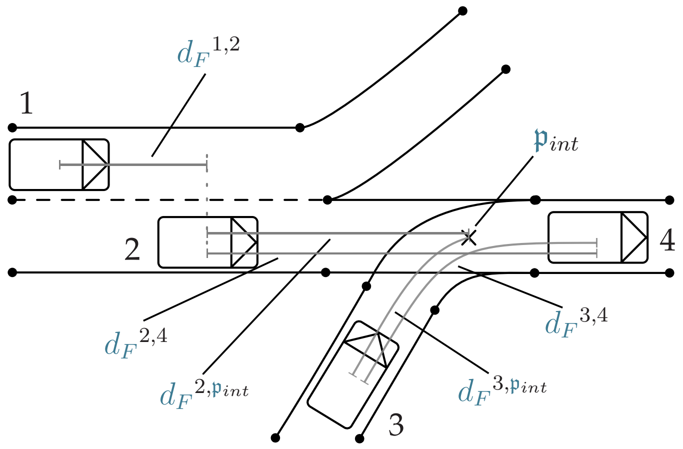


Figure 3.5: Schematic traffic with distances between traffic participants (Adapted from [7])

edge	κ_{rel}	d_F	d_{ip}
$e^{1,2}$	lat	$d_F^{1,2}$	-
$e^{2,1}$	lat	$-d_F^{1,2}$	-
$e^{2,3}$	int	-	$d_{ip}{}^{2,\mathfrak{p}_{int}}$
$e^{3,2}$	int	-	$d_{ip}{}^{3,\mathfrak{p}_{int}}$
$e^{2,4}$	lon	$d_F^{2,4}$	-
$e^{3,4}$	lon	$d_F^{3,4}$	-

Table 3.2: Edge distance attributes of the scene described in Figure 3.5

Figure 3.5 illustrates how distances are stored in the edges through a schematic example and the corresponding distances are listed in the Table 3.2. If two entities have a *longitudinal* relation (e.g. $e^{2,4}_{scene}$), the distance along the centerline of the road to the entity's projection is calculated. This ensures that any lateral shift of the entity does not affect the distance. The principle remains the same for merging lanes. It should be noted here again that different elementary road segments can also be combined to form a path, as long as they follow each other *consecutively* (see $e^{3,4}_{scene}$). The point where the two roads merge or overlap is called $\mathfrak{p}_{int}$ in this example. $\mathfrak{p}_{int}$ is located at the geometric center of the overlapping road segments. $\mathfrak{p}_{int}$ serves as an anchor to determine the distances of intersecting vehicles. If there exists an edge $e^{2,3}_{scene}$ with $\kappa_{road} =$ *intersecting*, only the distance of the tail node entity (in this example 1) to $\mathfrak{p}_{int}$ is defined. However, since all intersection edges have an antiparallel partner (here $e^{3,2}_{scene}$), the distance of the other traffic participant to the intersection point is taken into account in the inverse edge. At parallel adjacent roads, the distance between the two entities is determined similarly to longitudinal calculation. In this example, vehicle 2 is projected to the same lane as vehicle 1. Then the distance between both traffic participants is measured (compare $d^{1,2}_{F}$).

3.4 Implementation

The generation of the Semantic Scene Graph $G_{scene}(t)$ for a timestamp t from an object list consists of three steps.

For the calculation of the required distances and attributes, the open source libraries lanelet2 [127] or liblanlet [128] are used. The advantage of these libraries and HD-map of the used dataset are that the map is already divided into road segments called lanelets. In addition, the map is stored directly in a road graph and the calculation of relations as described Section 3.3.3 can be applied directly to it.

First, the distance d_n between all traffic participants $i \in I$ contained in the object list to all road segments is computed. A threshold value is specified in order to filter out non-relevant traffic participants ($I \rightarrow I', I' \subseteq I$), e.g. vehicles parked in parking lots or pedestrians who are far away. Subsequently, for all remaining $i \in I'$, all projection identities m^i are estimated. This is done by comparing the centerline of each road element of the road map with each pose (x^i, y^i, ψ^i) of $\mathcal{X}^i$ in $S_{\mathbb{R}^2}$. The resulting projected states $\hat{\mathcal{X}}^i$ are stored as node elements in a graph structure. For larger maps, a subdivision of the scene into sub-scenes is recommended. Since the approach presented here is developed for the consideration of small, local traffic scenes, such as an intersection or highway entrance, the spatial subdivision of the map is not further considered. In the second step, the relations between all traffic participants are determined. To do so, for all $\hat{\mathcal{X}}^i$ ($\forall m^i$), all potential paths $\mathcal{P}^i_{road}$ (routes the vehicle can take regarding its current position) are searched. Using a Dijkstra's algorithm [129], a set of possible paths $\mathcal{P}^i_{road}$ in G_{road} (all $e_{road} \in E_{road}$ should be treated as undirected edges, to also find reverse-edges) with a given starting node i are found. The sum of the lengths of the road elements is chosen as cutoff distance. Then, all projection identities m^i of $\hat{\mathcal{X}}^i$ are compared with all $m^j \notin \hat{\mathcal{X}}^i$. Assuming m^i_a and m^j_b have the possible paths $\mathcal{P}^i_{road}$ and $\mathcal{P}^j_{road}$; whenever $b \in \mathcal{P}^i_{road}$, a relation e_{scene} is created. κ_{rel} is determined using the conditions described in Section 3.3.3. In addition, to identify future intersecting paths, all road elements in $\mathcal{P}^i_{road}$ are compared with $\mathcal{P}^j_{road}$ and checked for the *overlapping* condition. If this statement is true for any element, then $\kappa_{rel} = \textit{intersecting}$.

To make the graph usable for other applications, it is exported to the dot language [130]. This interface enables data to be accessed easily from a wide variety of applications.

For further data processing, it often makes sense to convert this information into matrices. This is indispensable for creating datasets for machine learning approaches. For this purpose, the following file format, which is based on the TUDataset [131] is suggested for the graph information.

3.4.1 Graph Topology

It is common that connections between nodes are specified by an $n \times n$ adjacency matrix $\mathbf{A}$ ($n =$ number of nodes). If there exists an edge between node i and node j the entry at the i-th row and j-th column $\mathbf{A}_{ij} = 1$. In this application it makes sense to store the connectivity information in the coordinate format (COO) $\mathbf{A}'$, as this facilitates the description of parallel edges. $\mathbf{A}'$ is a $n \times 2$ matrix, where each row contains both nodes (v^i_{scene}, v^j_{scene}) which are connected by an edge e_{scene}.

$$\mathbf{A}'_k = [v^i_{scene}, v^j_{scene}], \tag{3.9}$$

where $k \in e_{scene}$, which also allows enumerating edges.

3.4.2 Node Attributes

The classification and the current state of a node i is specified in a vector u^i which is concatenated with all other vectors u of the nodes of v_{scene} which results in a $n \times \|u\|$ matrix $\mathbf{B}$. The first entries of u carry the information of the object's classification κ^i_{obj} in the shape of a one-hot encoding. The one-hot encoding function $\delta^{type}_{\kappa^i_{obj}}$ distinguishes between the classification type $type \in \{car, pedestrian, bike, truck, other\}$ of the object. If, for example, an object i is classified as a bike then $\delta^{bike}_{\kappa^i_{obj}}$ returns a 1:

$$\delta^{type}_{\kappa^i_{obj}} = \begin{cases} 1, & \text{if } \kappa^i_{obj} = type, \\ 0, & \text{else} \end{cases} \tag{3.10}$$

Furthermore, the absolute velocity $|\dot{x}_i, \dot{y}_i|$ of the object i is added.

$$\mathbf{B}_i = u^i = [\delta^{car}_{\kappa^i_{obj}}, \delta^{pedestrian}_{\kappa^i_{obj}}, \delta^{bike}_{\kappa^i_{obj}}, \delta^{truck}_{\kappa^i_{obj}}, \delta^{other}_{\kappa^i_{obj}}, |\dot{x}^i, \dot{y}^i|] \tag{3.11}$$

3.4.3 Edge Attributes

An edge's information is denoted by $\mathbf{A}'_k$ are specified by the edge attribute matrix $\mathbf{C}$ in the row k. Similar to the node attribute vector u, the edge attribute vector $\mathbf{C}_k$ of e_{scene} contains both classification and conditional information. The classification is described by a one-hot encoding defined by $\delta^{type}_{\kappa_{rel}}$ where $type \in \{lon, lat, int\}$ (compare Equation (3.10)).

$$\mathbf{C}_k = [\delta^{lon}_{\kappa_{rel}}, \delta^{lat}_{\kappa_{rel}}, \delta^{int}_{\kappa_{rel}}, P_{rel}, d_F, d_{ip}, a, d^i_n, \Phi^{ia}, b, d^j_n, \Phi^{jb}] \tag{3.12}$$

In certain applications, the use of parallel edges is not feasible due to mathematical or time constraints. Therefore, if necessary, all parallel edges between two nodes can be merged into a single edge. If two or more edges have differing d_F or d_{ip} values (e.g. node 1 and node 3, as illustrated in Figure 3.4), one value has to be selected. In this implementation, there are two methods by which an edge may be selected: either the edge with the highest probability P_{rel} or the edge with the lowest value for d_F or d_{ip}. All other edges with coherent information can then be consolidated.

$$\tilde{P}^{type}_{rel} = \frac{P_{rel}(\kappa_{rel} = type)}{\sum_{\kappa_{rel}} P_{rel}(\kappa_{rel})}, \quad type \in \{lat, lon, int\} \tag{3.13}$$

$$\tilde{P}^{lon}_{rel} + \tilde{P}^{lat}_{rel} + \tilde{P}^{int}_{rel} = 1 \tag{3.14}$$

This gives three probability values for each edge $(\tilde{P}^{lon}_{rel}, \tilde{P}^{lat}_{rel}, \tilde{P}^{int}_{rel})$, which are normalized, describing how likely it is that the two traffic participants have a certain relationship to each other. This would result in the following entries in C for the edges of the graph:

$$\mathbf{C}_k = [\tilde{P}^{lon}_{rel}, \tilde{P}^{lat}_{rel}, \tilde{P}^{int}_{rel}, d_F, d_{ip}, a, d^i_n, \Phi^{ia}, b, d^j_n, \Phi^{jb}] \quad (3.15)$$

If the annotation of the road map contains traffic rules, it is also possible to encode, if a traffic participant (or projection identity) has to give way to the other traffic participant. This, however, is not utilized in this thesis, as the INTERACTION dataset employed does not provide any of this information.

3.4.4 Execution Time

In many cases, a short execution time for the algorithm is crucial. The following section discusses the results of performance tests for the proposed framework. In Figure 3.6 the impact of both a growing traffic scene in context of more entities and a larger road geometry is shown. The blue plot is created artificially. Here, entities are randomly created on a static street geometry, which have several projection identities and as many relationships with each other as possible. The current implementation consists of 3 steps: Matching the objects to the road, where the number of road segments $\|V_{road}\|$ and the number of traffic participants $\|I\|$ have an impact on the computation time ($\mathcal{O}(\|V_{road}\| \times \|I\|)$). The second step is to find all future paths using Dijkstra's algorithm, which depends on the size of the map ($\mathcal{O}(\|V_{road}\|^2)$). The final and most computationally intensive step is the comparison of the resulting projection identities $\|m\|$ of a traffic participant with all other traffic participants and their corresponding projection identities ($\mathcal{O}(\|m\|^2 \times \|I\|^2)$).

Although finding paths within the road network has a high time complexity in theory, in practice the road graphs are not very interconnected, which makes the calculation very efficient. This timing factor can be neglected in contrast to the number of entities in the application. Most intersections of the evaluated scenarios of the INTERACTION dataset consist of between 14 and 100 elementary road segments. For larger maps, however, it is suggested to divide them into sub-maps.

The number of the entities in a single scene on the other hand limits the applicability of the framework in the context of real time usability. The current implementation indicates that the number of traffic participants and the corresponding number of projection identities have a significant impact on the computation time. If one assumes a repetition rate of the incoming object lists of 10 Hz (frame rate of the evaluation dataset), the calculation time should be less than 100 ms. This means on average there should not

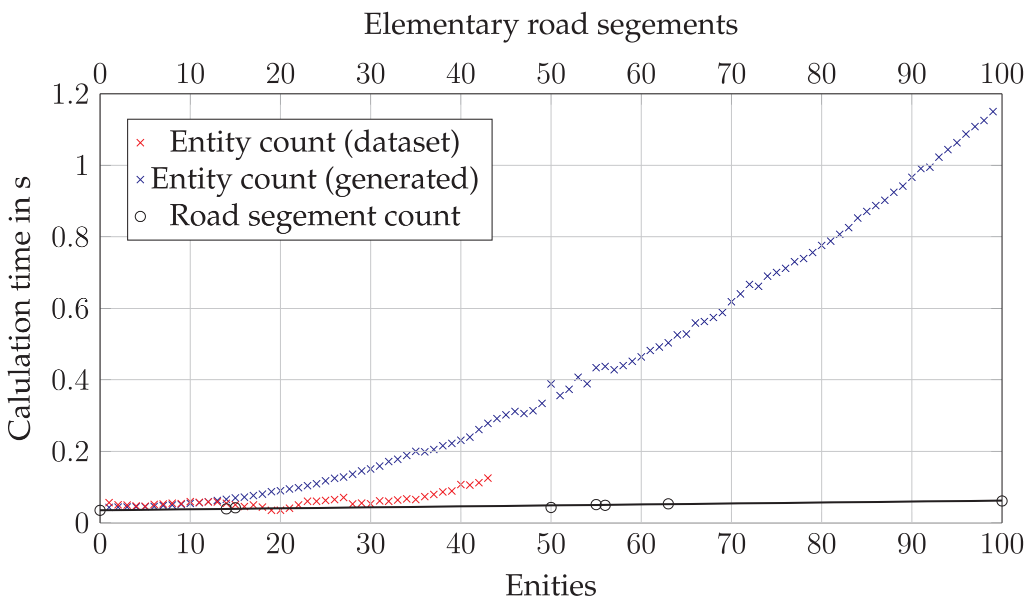


Figure 3.6: Impact of the count of entities in a traffic scene with a fixed road geometry on the calculation time (blue, red). Impact of a larger road graph with a fixed number of entities (black). (Adapted from [8])

be more than 20[*] entities in the scene to retain a lag-free processing (see Figure 3.6). This, however, is tested for the worst case, where the scene graph is quite dense, and each entity has several projection identities. The evaluation dataset contains scenes with up to 43 traffic participants. In general, the average calculation time is significantly lower than the worst case time (compare blue and red plot in Figure 3.6). The highest value with 125 ms is for scenes with 43 traffic participants. Assuming that the dataset represents reality well, the limit of 100 ms set above would be fulfilled on average for all scenes with less than 40 traffic participants.

3.4.5 Completeness and Scalability

In this section, it will be discussed when the SSG can be applied and when the model is incomplete.

[*]The framework was run on a single core of an Intel® Core™ i7-8750H and was not optimized in terms of parallelization and memory requests, which could lead to a significant improvement

The SSG is applied to the entire INTERACTION dataset. A key feature of this dataset is that different intersection types and thus different road geometries are included.

	Intersection	Roundabout	Merging
Nodes	9.8	6.9	14.5
Edges	69.6	46.5	111.3

Table 3.3: Graph statistics of different intersection types

Table 3.3 shows the average nodes and edges per scene of the SSG for each intersection type in the dataset. Here, differences between the intersection types can be seen. Roundabouts have on average a smaller number of edges and nodes in the graph. In intersections, parallel lanes exist more often and thus traffic participants can have more relationships with near vehicles. Most of the merging scenes of the dataset included scenes from a multi-lane roadway. Thus, many lateral relations are included and SSGs contain on average more edges per node.

In principle, the average number of nodes can only indicate the size of the traffic scene, but it cannot make a concrete difference about how crowded or wide an intersection area is.

In addition, the completeness of the scenes is evaluated. The number of traffic participants in the original scene is compared with the number of nodes in the corresponding SSG. In the examined dataset, 79.9 % of the scenes are depicted in their entirety. This means that every traffic participant present in the scene $S_{\mathbb{R}^2}$ can be mapped in the G_{scene}. However, the individual exceptions indicate the weaknesses of the scene graph model: In Figure 3.7 an invalid traffic scene is depicted. Here, the marked vehicle is not translated to the SSG. The reason for this is that this vehicle is too far away from the actual road and therefore cannot be assigned to any lane (see Equation (3.5)). In general, the matching distance could be increased, but this would result in too many irrelevant lanes being taken into account for other vehicles, which would increase the calculation time and also make the graph too confusing. Such scenes can be caused by tracking errors, missing annotation of the road (probable cause of the scene in Figure 3.7) but also by behavior of traffic participants that do not comply with traffic rules. An example for this can be an obstacle on the roadway that can only be avoided by changing to the sidewalk or the oncoming lane.

As a consequence, the model can only describe traffic scenes that correspond to the traffic rules. The corner cases described above cannot be described

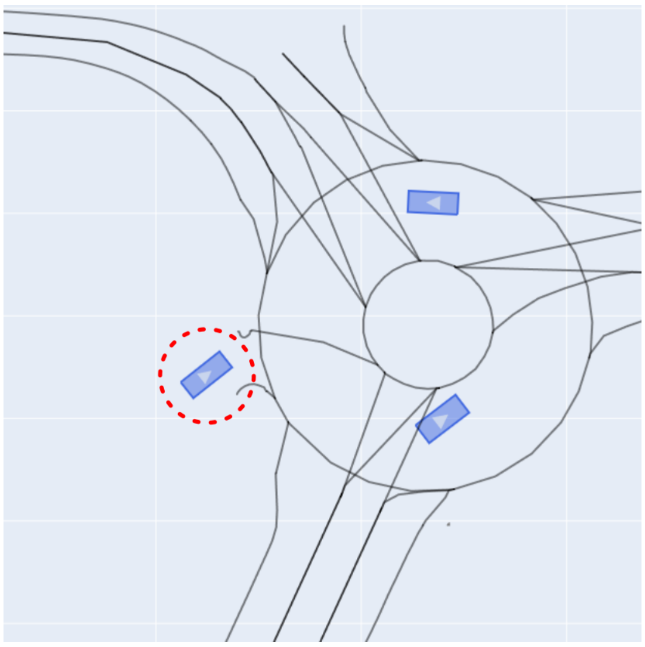


Figure 3.7: Invalid traffic scene: marked vehicle is outside of lane boundaries.

correctly in the SSG. A temporal consideration of several successive scenes, however, could improve the completeness.

3.4.6 Situation-specific Analysis

In this section, the generated Semantic Scene Graph are discussed in depth using two exemplary scenes.

Figure 3.8 displays two traffic scenes in both a scene graph representation and a birds-eye view (traffic scene in the Cartesian space). To enhance the clear visualization, the two corresponding antiparallel edges between two entities are represented in this figure as a bidirectional edge. Additionally, the cutoff distance is set to 30 m. The first traffic scene (Figure 3.8a) shows several vehicles entering a roundabout. On the right-hand side, five vehicles are following each other, with the first vehicle having just entered the roundabout. The graph highlights the vehicle group {13, 14, 15, 17, 18} with a darker blue color. This group can be separated in the graph by checking only for successive *longitudinal* relations. The vehicle with the ID 10 should technically be added to this group. However, since this vehicle has no other *intersecting* relation with group {3, 16}, it can be considered an extra group. This allows relevant traffic participants to be clustered only by the way they relate to each other, for example to distinguish between different traffic situations [72]. Figure 3.8b shows a second traffic scene at a different place with different road geometry. Although the scenes appear different at first impression, both in terms of the geometry of the road and the position of the traffic participants. However, a closer examination of the graphs reveals similarities. Both graphs depict three distinct groups of traffic participants: those entering the intersection area (darker blue), those crossing the area (yellow), and those who have already left the area (light blue). This example demonstrates how the SSG's generic description can be used to compare two different traffic scenes. It can be shown that the topology of the traffic scene is transferred in a certain form into the graph structure. It is important to note that the traffic scene is described solely by the interaction between different traffic participants and their current motion state. Therefore, this scene description cannot be used to explicitly describe special road conditions, such as geometry, or blind spots. An in-depth and quantitative discussion of the sub-grouping of situations within a traffic scene on the basis of the SSG is not considered further in this work. However, the result of the qualitative analysis considered in this section serves as the basis for the scene-based clustering method in Chapter 5.

3.5 Summary

This chapter presents the SSG, addressing the question raised in Research goal 2. The graph model presented here depicts relationships between traffic participants through the use of three distinct edge types. An edge relationship is defined as either longitudinal, lateral, or intersecting. Furthermore, distances between the respective vehicles and other parameters are stored within the edges to minimize the loss of information. This approach preserves relevant relationships between traffic participants that are defined by the road topology, while the road geometry is largely excluded from the description model. The result is a description model that condenses a traffic scene into relationships between the individual traffic participants, which in turn enables the description of their respective constellations. This suggests for a direct comparison of traffic scenes and traffic situations regardless of the recording location, as well as the exact position of individual traffic participants.

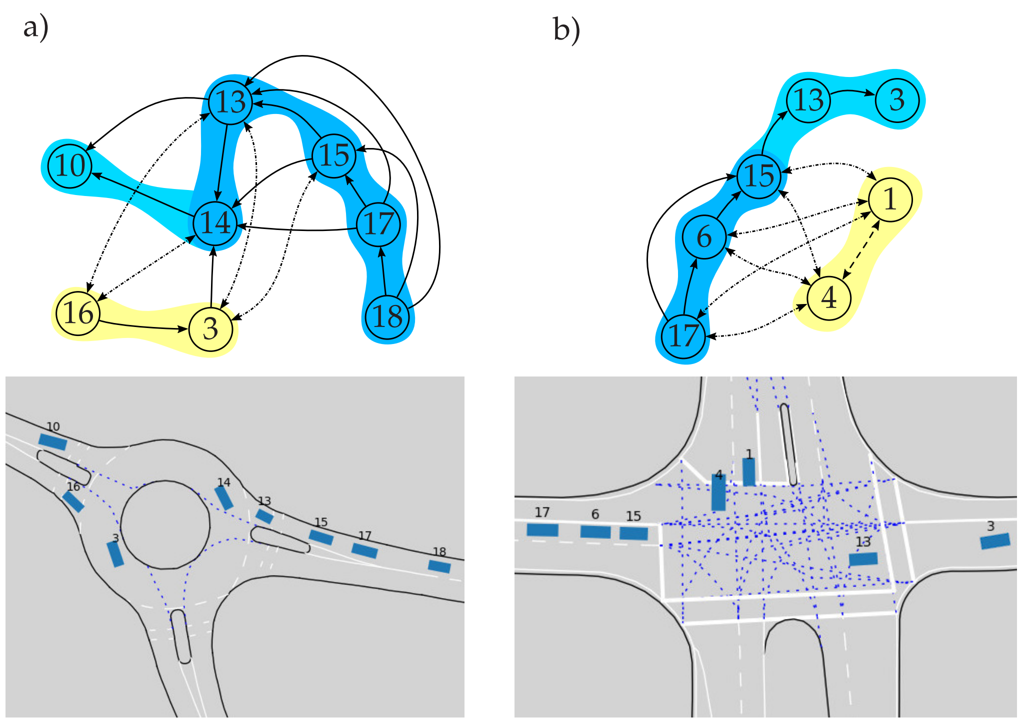


Figure 3.8: Two exemplary traffic scenes at different road geometries in the Cartesian space (bottom) and the corresponding SSG (top). Nodes are highlighted to improve the visualization of groups. (Adapted from [8])

4 Criticality Assessment

As previously discussed in the introduction, scenarios representing particularly challenging and critical traffic situations are of particular interest for scenario-based testing. However, in order to identify relevant scenarios and scenes, an evaluation method is required to sort them according to relevance. The evaluation of a scene or scenario can vary depending on the subsequent use case. In this work, mainly scenes and scenarios in which traffic participants are harmed or nearly harmed are considered critical (see Section 2.1.1). Consequently, taking into account the ODD, these are collisions and near-collisions between traffic participants.

Quantitative descriptions are obtained by the use of criticality metrics [59, 132, 43]. These metrics aim to determine whether a given scene or scenario includes hazardous elements. The advantage lies in the ability to use a key metric for a direct assessment of scenario criticality, enabling automated annotation on a large scale. Furthermore, most metrics are agnostic to the SuT and can evaluate it objectively. Nevertheless, a challenge arises as many criticality metrics tend to only focus on specific characteristics. Additionally, not all metrics are universally applicable to every scene, often requiring specific conditions. Moreover, assessments sometimes concentrate solely on complete scenes or isolate individual elements within a scene.

One of the main aims of this evaluation methodology is to reduce the restrictions on the applicability of scenario types that can be observed in many conventional measurement methods [59]. Within this section, an assessment methodology is introduced for the comparative analysis of traffic scenes using diverse metrics derived from motion datasets (see Section 3.1). The evaluation encompasses an exploration of how various established metrics are impacted by different types of traffic scenes, proposing enhancements. By combining and expanding universal metrics into a comprehensive scene profile, the objective is to facilitate real-world applicability and address associated challenges. Simultaneously, this evaluation of traffic scenes establishes a baseline against which different methodologies, as demonstrated in this thesis, can be compared or contextualized (refer to Section 7.4).

Parts of this chapter have been published previously in the following peer-reviewed publications [6], and some of the content has been adopted verbatim.

4.1 Related Work

Criticality metrics have long been used in the context of traffic, and various types have been developed for different applications, particularly for testing highly automated vehicles. Schütt et al. [133] propose a taxonomy that categorizes metrics into different domains, such as model quality, system under test quality, or scenario quality. Metrics in these domains can be divided into different levels of resolution. The highest resolution is called *nanoscopic* and evaluates the quality of individual entities for a single time step. The second level is called the *microscopic* level and evaluates the quality of time histories of a single entity. At the *macroscopic* level, which has the lowest resolution, the quality of sets of scenarios or coupled systems is described. One criticism of this taxonomy is that it focuses exclusively on an ego-vehicle. It does not take into account how many traffic participants are included in the metric. Since in the course of this work a scene and potentially several traffic participants are to be evaluated in a composite manner, a quantitative categorization is used in addition to the temporal categorization. For this purpose, the terms *individual* metric, *binary* metric, and *poly*-metric are introduced. The individual metric refers to parameters that affect a single traffic participant, such as speed. Binary metrics describe the relationship between two traffic participants. Poly-metrics include several or all traffic participants within an area or the entire scene. Most metrics adopt a binary nature, measuring only between two traffic participants. They convey the severity of a situation between these two entities. However, crucial situations may go unnoticed if certain actors within a scene are omitted. Binary criticality metrics are cost-effective because they are easily computed and require only partial information about a given scene.

Another way to categorize metrics is by their use of spatial attributes, temporal attributes, or other indicators. Mahmud et al. [134] categorize metrics based on temporal-based proximity indicators, distance-based proximity indicators, deceleration-based indicators, and other indicators. Criticality metrics are commonly based on temporal or spatial proximity, assuming that the closer two traffic participants are, the more likely they are to collide. These metrics typically indicate a conflict if the resulting value is below a specified threshold.

In general, there are a multitude of metrics (compare the work of Westhofen et al. [59]) that are not described in this chapter. The focus here is on the most common metrics that can be used to make a certain statement about a dangerous situation, and the metrics that can be calculated on the basis of the information in the dataset used (refer to Section 3.1).

4.1.1 Spatial Metrics

The Euclidean Distance (Dist) is widely used in various applications because it represents the straight-line geometric distance between two points in space. While this simplicity can aid interpretation, it has limitations, as it does not account for the direction or orientation of vehicles. In traffic scenarios, where changes in direction and orientation angles are crucial, the Dist may not provide accurate assessments. This is particularly true when individual vehicles deviate significantly from the average traffic situation. Moreover, criticality in traffic situations may not always have a linear relationship with distance. There are cases where a small distance is considered critical, while a large distance may not be. Therefore, relying solely on Euclidean Distance may not capture the complexity of assessing criticality in traffic scenarios.

For the Trajectory Distance (TD), the paths of the traffic participants are treated as curves in 2D space. This metric measures the distance along intersecting trajectories between two traffic participants, providing a deeper insight into intersection and merge scenarios. It should be noted that this metric can only be evaluated in a retrospective manner (see also Post-encroachment-time), since the actual trajectories that the vehicles will follow are not known in advance with certainty. Besides, it has the same drawbacks as the Dist.

4.1.2 Temporal Metrics

Time To Collision (TTC) is a widely used metric in traffic safety analysis, that offers a comprehensive perspective on collision risks within a traffic scene [135, 136]. It is defined by Equation (4.1), where d_F is the distance along the lane between a leading and the following vehicle, where ν^l and ν^f are the respective velocities.

$$\text{TTC} = \frac{d_F}{\nu^f - \nu^l} \tag{4.1}$$

TTC's predictive value stands out as it furnishes a measure of time it would take for two vehicles to collide, assuming their present speed and trajectory persist. This predictive nature makes TTC a valuable tool for assessing potential collision risks in varying scenarios. TTC is based on quantitative attributes such as current speed and distance between vehicles, rendering this metric generally applicable. This objectivity enhances its utility by facilitating straightforward analysis and comparison across different scenarios. The popularity of TTC is due to its accessibility to both experts and non-experts, which facilitates effective communication and public awareness of collision risks. It provides a direct indication of the time until a potential accident may occur if no action is taken to change behavior. Furthermore, the TTC algorithm enables real-time assessment by utilizing information from vehicle sensors to dynamically calculate collision risks as the traffic scene unfolds. This real-time capability enhances its relevance in dynamic and evolving situations. However, it is crucial to acknowledge the limitations of TTC. The algorithm operates under the assumption that vehicles will maintain a constant speed and trajectory, which may not align with the unpredictability of real-world scenarios. TTC predictions can be significantly impacted by sudden changes in speed or direction. Furthermore, TTC is a one-dimensional metric that is limited to car-following situations on the same track, as it does not take into account lateral spatial information. The limitation of Equation (4.1) is that once ν^f and ν^l are equal, the TTC becomes undefined. To address this problem, the inverse Time To Collision (TTC*) was introduced (see Equation (4.2)). A higher value indicates a more critical situation.

$$\mathrm{TTC}* = \frac{\nu^f - \nu^l}{d_F} \tag{4.2}$$

The TTC metric serves as a foundational framework for various criticality metrics, such as the Potential Time To Collision (PTTC) [137] and the Worst Time To Collision (WTTC) [138]. TTC assumes that all current conditions will persist, while PTTC takes into account potential changes in the leading vehicle's trajectory, providing a more dynamic assessment of collision risk. Specifically, it assumes that the vehicle in front brakes with constant acceleration a^l, while the following vehicle continues to travel at a constant velocity as shown in Equation (4.3).

$$\mathrm{PTTC} = \frac{-(\nu^f - \nu^l) \pm \sqrt{(\nu^f - \nu^l) + 2a^l d_F}}{a^l} \tag{4.3}$$

WTTC's primary objective is to minimize the occurrence of false negatives in identifying critical traffic scenarios by estimating the criticality of the most severe conceivable situation based on the initial scene. This metric quantifies the time required for vehicles in a given traffic scenario to reach their closest point of approach. The calculation takes into account two main factors: the distance to the closest point of approach between two vehicles and their relative velocity. The distance to the closest point of approach is determined as the minimum separation between the paths of the vehicles, while relative velocity accounts for both the speed and direction of the vehicles in relation to each other. The WTTC shares many of the strengths and limitations of the TTC, although the interpretability of the WTTC may not be as straightforward. Notably, the WTTC holds an advantage in its applicability to various traffic scenarios, encompassing crossing situations and approaching vehicles. For the calculation of the WTTC, please refer to the corresponding paper [138].

The metrics mentioned above are commonly used for traffic scenarios on motorways or rural roads. However, metrics like Post-encroachment-time, Encroachment-Time or Gap Time are only suitable for intersection scenarios (primarily found at junctions or roundabouts).

Post-encroachment-time (PET) [139] is a criticality metric that calculates the duration between two actors passing through a given point or area of intersection. This metric offers valuable insight into the temporal dynamics of vehicle interactions. It provides a comprehensive understanding of the time that has elapsed after a vehicle has crossed a mutual intersection point, which will be passed by another vehicle. By assessing the post-intersection time, one can gain a deeper perspective on the potential consequences and accidents of vehicle movements in close proximity. It is important to note that PET can only be applied retrospectively to a scenario and cannot be applied to the online evaluation of a system. Allen et al. [139] adapt the idea of the PET and use a constant velocity model to calculate the estimated time between two vehicles crossing the conflict zone, which he calls Gap Time (GT). GT is an approximation of the PET but can be determined online before the actual crossing of the conflict zone.

Encroachment-Time (ET), on the other hand, focuses on the duration that the first actors remains within the boundaries of the intersection. This metric measures the duration of a vehicle's presence in a potential conflict zone with another traffic participant. It is important to note that this value does not necessarily correlate with the criticality of the situation, as it does not take into account any dynamic parameters of the traffic participants involved [139].

4.1.3 Others Metrics

Hallerbach et al. [140] introduce a methodology for assessing the criticality of traffic on a specific highway section. Unlike conventional approaches that focus only on the ego-vehicle and one other traffic participant, this method extends its scope to include different domains of interest. These domains include larger road sections, traffic around the ego-vehicle, and the ego-vehicle itself. The traffic quality assessment involves a four-part calculation, each with uniquely weighted coefficients that contribute to an overall criticality score. The first component, referred to as the macroscopic metric, quantifies traffic density by considering traffic flow rate and average travel speed. The second component, the microscopic metric, takes into account the speed deviation and average speed within the highway section around the ego-vehicle. The third aspect of traffic quality assessment, the nanoscopic metric, focuses on close-range interactions within a smaller radius around the ego-vehicle. This metric calculates criticality based on speed deviation and average values within this defined radius. The fourth and final component, the individual metric, focuses on the mean velocity and standard deviation of acceleration of the ego-vehicle. The Inverse Universal Traffic Quality (TQ), which is applied in the latter course of the work, is an enhancement of the traffic quality [140] proposed in the work of Schütt et al. [10]. The improved TQ omits learned coefficients between the individual stages and can therefore be used universally. Furthermore, the direction of travel of all vehicles in a scene is no longer limited to just one direction [10]. Note that in contrast to the highway traffic quality, the higher the value of TQ is, the more critical the scene becomes.

The Safety Force Field (SFF) model, as introduced by Nister et al. [68, 141], is designed to enhance safety by preventing collisions. It achieves this by ensuring that the ego-vehicle does not contribute to an unsafe environment, recognizing that a guarantee of a secure journey cannot be provided when other actors share the road. Inspired by the RSS model proposed by Shalev-Shwartz et al. [34], the SFF serves as a framework for assessing the driving situation in which a highly automated vehicle is currently operating. The goal is to enable the vehicle to make safe decisions based on this model. The SFF takes into account the states of all traffic participants and predicts their behavior in the near future to provide a comprehensive assessment of the situation. To accomplish this, the model computes the claimed set, which represents the space that a vehicle is expected to occupy over time, resulting in the Safety potential (SP). By analyzing the geometric overlaps of the occupied sets of various traffic participants, an optimal safety procedure can be computed. This approach facilitates a thorough consideration of the dynamics of the traffic environment, allowing for effective decision-making and contributing to overall road safety.

The advantage of these two metrics is that they can be applied regardless of the scene type (e.g. intersection or freeway scenarios) and return a valid value. Nevertheless, the geometric calculation of the SP is particularly computationally intensive in comparison to the Dist or TTC.

4.2 Experiments and Evaluation

The metrics (ET, PET, WTTC, Dist, PTTC, TTC, GT, TD, TQ, SP) of the previous Section 4.1 are implemented and calculated independently of each other. Due to the time-consuming computation of some metrics and the fact that some metrics require knowledge of the scenario after the fact (see PET, ET, TD), the computation of all metrics in their full depth can only be done offline, i.e., in the case of microscopic metrics, after the entire scenario has been recorded. However, the application of the framework is the general evaluation of the dataset on the one hand and the final evaluation of the scenarios on the other hand. The metrics implementation is publicly available on GitHub[*].

Depending on the metric type, several values may be calculated for a scene. Whereas binary metrics calculate one value for each different entity pair (ET, PET, WTTC, Dist, PTTC, TTC, GT, TD), poly-metrics only calculate one value per traffic participant (SP) or per scene (TQ). A permutation-invariant function is required to summarize a varying number of values (e.g., max, mean or min operator). Since the primary objective is criticality, the most critical value of all traffic participants in a scene is evaluated. For instance, the Dist is evaluated using the min operator, while the TQ is evaluated using the max operator.

Most distance- and time-based metrics tend to approach zero in critical scenarios. For instance, a PTTC $= 0.0\,\mathrm{s}$ signifies a collision occurrence. Additionally, the majority of metrics exhibit diminishing significance as their values increase. For example, if the PTTC exceeds $15\,\mathrm{s}$, $30\,\mathrm{s}$, or even $50\,\mathrm{s}$, it does not provide additional insights into the criticality or near-crash situations involving two actors in a scene. All three time values categorize a scene as non-critical, as substantial events can still unfold within these time intervals. To facilitate a visual comparison (see Figure 4.2), the following function to invert and scale the metrics is employed. This is necessary in cases where the metrics do not inherently yield values between 0 (non-critical) and 1 (critical):

$$f(x) = e^{\alpha'(-x)} \tag{4.4}$$

Here, x represents the measured metric value, and α' serves as a coefficient reflecting sensitivity. Coefficients, such as $\alpha' = 2.0$ for reduced sensitivity or $\alpha' = 0.5$ for increased sensitivity, can be employed depending on the desired metric characteristics. In the following examples, the value of $\alpha' = 1.0$ was chosen in order to achieve a balanced sensitivity.

[*]https://github.com/fzi-forschungszentrum-informatik/scene-fingerprint

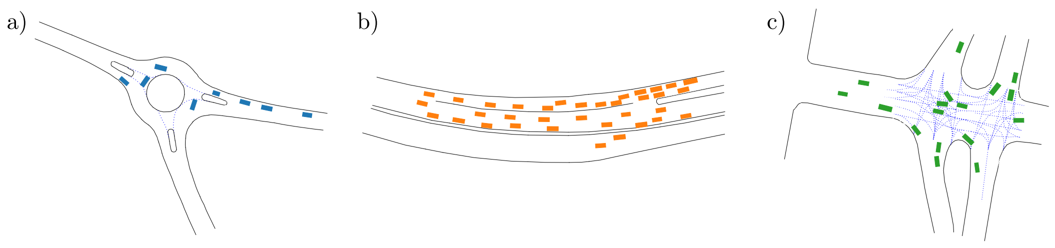
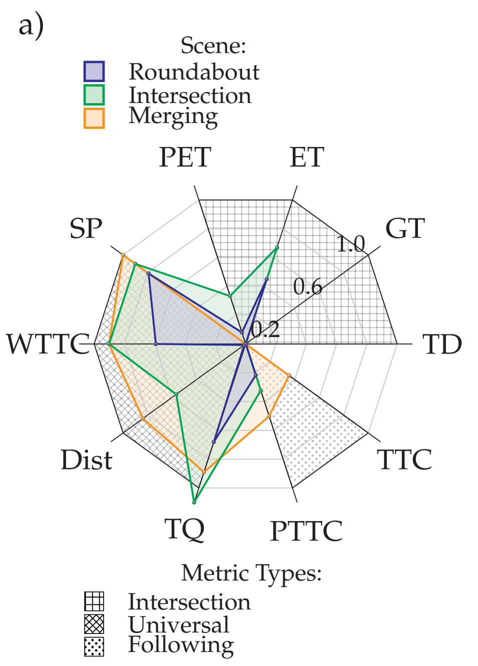


Figure 4.1: Scenes from different Scenarios: roundabout (blue traffic participants, (a)), merging (orange traffic participants, (b)), intersection (green traffic participants, (c)) [6]

In order to provide a comprehensive assessment, three distinct scene types are employed as an illustrative example. These encompass a roundabout scenario, a merging scenario on a highway, and an urban intersection. The first scene (Figure 4.1a) takes place at a roundabout in Germany. Apart from the three vehicles entering the roundabout, most of the cars are stationary, waiting for a chance to enter or slow down to avoid a collision. The merging scene (Figure 4.1b) takes place on a Chinese highway with high traffic density and limited space around some actors, making it a challenging situation. In the third example (Figure 4.1c), a US intersection traffic scene is depicted, which Zhan et al. [85] classify as a high-risk situation. Three cars traveling in opposite directions are crossed by a car passing between them. Most of the other vehicles depicted in the traffic scene are not critical, as most of them are stationary and waiting to enter or approach the intersection.

The qualitative evaluation of the three different example scenes of the presented illustration using Kiviat diagrams offers a variety of analytical options. The combination of various measures of criticality in a single visual representation is here referred to as the scene's fingerprint. [6].

Figure 4.2a displays a diagram of the proposed metrics for all three exemplary scenes at three different road geometries. From a qualitative point of view, the larger the area spanned between the individual metrics, the more critical the scene. In the merging scenario, data points are missing for all metrics developed for intersection scenarios. Therefore, it can be assumed that there is no situation with overlapping trajectories between any two vehicles in this scene. The depiction of this exemplary traffic scene (Figure 4.1b) shows that no vehicles have the ability to intersect the trajectory of other vehicles and confirms the statement indicated by the metrics. Furthermore, it becomes clear why the following vehicle metrics, apart from the universal metrics, are particularly significant. Due to the closely following vehicles, which



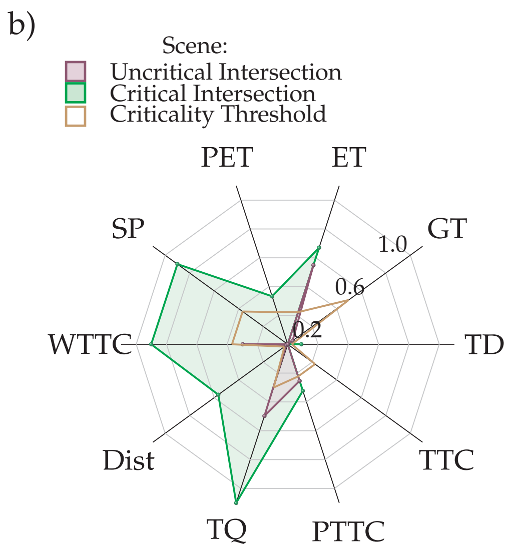


Figure 4.2: Comparison of criticality metrics (TQ, TD, GT, ET, PET, SP, WTTC, Dist, TTC) of the scenes in Figure 4.1 (a). Comparison of a critical (green) and an uncritical (violet) scene (b). (Adapted from [6])

drive behind one another in vehicle convoys, the safety distance tends to be particularly small.

In the roundabout scene (blue) from Figure 4.1a only values for PET and ET are able to be calculated. This indicates that the trajectories of different traffic participants intersect in this scenario. Again, it should be noted that the two metrics are temporal microscopic metrics, which means that they are not calculated based on a scene, but on the entire course of the scenario. Accordingly, it is not necessarily visible in the image of the scene under the vehicle constellation shown, and either happened chronologically before the scene shown, or will happen in the near future. Overall, the intersecting metrics show relatively small areas, suggesting that the scene is not critical. The last scene is a typical multi-lane intersection (green) and is shown in Figure 4.1c. It is rated as more critical than the roundabout scene, based on the examined metrics.

Figure 4.2b shows in green the metrics for the scene (scenario) in Figure 4.1c. In addition, metrics of another scenario (violet) that takes place on the same map at a different time are shown. In the second scenario (not shown), five vehicles drive relatively far apart on the intersection. This scenario shows a low criticality for all metrics, except for the ET value, which is well below the threshold value. The brown line indicates a criticality threshold of the individual metrics. The following thresholds are used: Dist 5 m, GT 0.5 s [142], TQ 1.2 (derived from evaluations in [10]), PTTC 1.5 s, ET 1.5 s, PET 1.5 s [139], TTC 1.5 s [136], WTTC 0.7 s. All of the aforementioned threshold values have been scaled in a uniform manner, according to Equation (4.4).

Not all scenes have data values greater than zero for each axis. There may be two reasons for this: either the metric is not critical in that scene, or that specific metric does not support the type of scene being analyzed. In general, most metrics have been designed for specific purposes and scenario types. This is visualized as an example in Figure 4.2a using the metric types (Intersection, Universal, Following). However, several scene metrics can not be calculated during the dataset evaluation. Table 4.1 presents a comparison of various metrics and their respective positions within the complete dataset. The table illustrates the frequency of successful metric calculations. Specifically, the analysis focuses on determining the success of calculating the criticality of scenarios or scenes present in the dataset. Success in this context implies that for nanoscopic, binary metrics, the relevant metric can be computed for at least one vehicle in the scene. For temporal microscopic metrics (ET, PET, GT) within the defined scenario, the metric can be computed for at least one pair of the vehicles.

The metrics labeled as universally applicable in Figure 4.2a demonstrate considerable versatility, as they can be calculated across nearly all of the scenarios examined. The TQ metric even satisfies the calculation conditions in every scenario without exception. In the case of Dist and WTTC, those scenes in which only a single traffic participant is visible, and therefore no distance can be calculated cannot be applied. Even the TTC, which is designed exclusively for following vehicles, can be calculated in the majority of intersection and roundabout scenarios. This is mainly due to the fact that many scenarios have a large number of traffic participants and at least two of them make a following trip. It is particularly interesting that the intersection metrics (PET, ET, GT, TD) can be calculated extremely rarely. This may be due to the way the metrics are calculated. There is the condition that trajectories or paths must intersect at one point and not overlap further. This neglects all merging scenarios and, for example, roundabout scenarios where vehicles drive behind each other for a given time on the roundabout.

Due to the limited applicability of the intersection metrics, they are not be considered further in the following part of this thesis. In addition, although ET and TD provide additional information for special traffic scenarios, the informative value of both is very limited in the context of criticality assessment of traffic scenarios. Furthermore, SP, although it provides a universal insight into various traffic scenarios, requires a large computational effort to determine compared to the other metrics. Although the application in this work is offline and therefore not necessarily time constrained, the current implementation of the computation of SP cannot be used for a large-scale application.

Location	Dist	GT	TD	PET	ET	PTTC	TTC	TQ	SP	WTTC
Merging MT	0.95	0.31	0.47	0.47	0.48	0.88	0.59	1.0	1.0	0.94
Roundabout LN	0.93	0.06	0.09	0.09	0.02	0.55	0.39	1.0	1.0	0.92
Roundabout SR	1.0	0.22	0.33	0.33	0.23	1.0	0.99	1.0	1.0	1.0
Roundabout EP	1.0	0.08	0.12	0.12	0.24	0.99	0.96	1.0	1.0	1.0
Intersection EP0	0.99	0.07	0.08	0.08	0.23	0.94	0.85	1.0	1.0	0.99
Intersection EP1	1.0	0.11	0.16	0.16	0.35	0.99	0.98	1.0	1.0	1.0
Intersection GL	1.0	0.27	0.35	0.35	0.36	0.99	0.93	1.0	1.0	1.0
Intersection MA	1.0	0.24	0.28	0.28	0.28	0.99	0.88	1.0	1.0	1.0
Intersection EP1	1.0	0.47	0.61	0.61	0.35	0.99	0.89	1.0	1.0	1.0
Total	**0.99**	**0.2**	**0.28**	**0.28**	**0.28**	**0.92**	**0.83**	**1.0**	**1.0**	**0.98**

Table 4.1: Applicability of the different metrics in relation to the different road types of the analyzed dataset.

4.3 Summary

This chapter presents several criticality metrics. They have been categorized into different classes depending on which aspects of a traffic scenario or traffic scene can be described by them. Subsequently, six criticality metrics are identified that can be used to classify scenes based on their criticality, taking into account different aspects. This so-called fingerprint of a traffic scene is the first component to answer Research goal 4. The fingerprint provides the basis for a generalizable evaluation key that can be applied to all traffic scenes. The aim of using these metrics for the evaluation of an initial scene is further pursued in Chapter 7. The criticality metrics are employed to establish a measure that offers an indication of the relevance of those scenes.

5 Scene Clustering

As already stated in Section 2.1.4, an efficient data-driven scenario testing requires evaluating and assessing test scenarios in advance to determine if they potentially trigger errors in the SuT. To achieve this, it is necessary to identify relevant scenarios or relevant scenes from a huge amount of data. To do so, the data must be clustered into groups. This process enables the identification of similar scenes that can address specific aspects of a HAV. For example, it may be used to focus testing on a particular situation. Alternatively, it can be used to define representative scenes for a cluster, speeding up testing. The prevalent approach is to manually label scenarios, which can then be grouped together. To avoid time-consuming manual labeling, especially with large amounts of data, the clustering should be done automatically.

In order to be able to cluster traffic scenes, the first step is to find a way to make different scenes comparable with each other by making relevant properties available through an appropriate description model [71]. For this purpose, the Semantic Scene Graph (SSG) comes into play as a robust interface to provide relevant scene information in a generalizable form for clustering (refer to Section 3.4.6).

In this chapter, a GNN-based contrastive learning approach is presented to automatically cluster traffic scenes described with the help of the SSG.

Parts of this chapter have been published previously in the following peer-reviewed publications [9, 4], and some of the content has been adopted verbatim.

5.1 Related Work

5.1.1 Clustering of Traffic Scenes and Scenarios

In recent years, numerous publications have emerged, addressing the clustering or categorization of traffic scenes. In this section, recent works are divided into two different categories: Feature-Based Approaches, where various attributes and time series of features are compared, and Maneuver-Based Approaches, which extend the analysis to utilize vehicle trajectories and movement patterns as key inputs for characterizing traffic scenarios.

Trajectory-Based Approaches

Bernhard et al. [143] employ the bag-of-words method to convert trajectories into histograms within spatially divided clusters. Scenario affinity is determined by the similarity of corresponding histograms. Another approach by Ries et al. [57] utilizes Dynamic Time Warping (DTW) to compare trajectories of traffic participants. Scenarios are considered as similar and can be grouped together when they involve the same types of traffic participants and have comparable trajectories. In addition to the DTW, the Hausdorff distance can be used to directly compare trajectories, which then enables clustering of trajectory data [144]. Harmening et al. [56] present two autoencoder-based models that reconstruct traffic scenes based on a grid map and traffic scenarios, utilizing temporally related position and speed features. In both cases, the embedding space is leveraged for clustering. Wurst et al. [145] propose a machine learning-based method for grouping similar scenarios and concurrently performing novelty detection. Road geometry, represented by a bird's-eye image, and road topology, constructed as a graph, are utilized to extract samples for a contrastive learning approach. In the study by King et al. [146], the road topology is also included in the analysis of the trajectory data. Furthermore, the trajectories of other traffic participants are compared with the trajectory of the ego-vehicle. By analyzing the maneuvers recorded in a dataset, trajectories can be classified into different logical scenarios and assigned to groups.

The survey conducted by Bian et al. [147] provides a more in-depth discussion of approaches and mathematical methods for comparing and grouping trajectories than the aforementioned representatives. The focus of the survey paper is not solely on the clustering of traffic scenarios; it discusses much more fundamental procedures.

Feature-Based Approaches

Kruber et al. [148] utilize features of a highway scenario concerning an ego-vehicle, such as distances to other traffic participants and velocities, employing a random forest approach to group similar traffic scenarios. In the work of Kerber et al. [55], traffic scenes are longitudinally compared using an eight-car neighborhood model, assessing the similarity of highway scenarios. Pairwise scenario comparisons are conducted based on the distances of other traffic participants within the eight surrounding areas around the ego-vehicle. Meaningful comparisons are restricted to scenarios with identical locations and the same number of traffic participants. The resulting clusters encapsulate similar maneuvers executed by both the ego and surrounding vehicles. Hauer et al. [54] adopt a similar concept, employing DTW to compare time series of multiple features. To reduce dimensionality, Principal Component Analysis (PCA) is applied for data projection into a clustering space.

5.1.2 Graph Clustering

A variety of architectural approaches can be employed to encode graphs for clustering purposes. One such option is the utilization of Graph (Variational) Autoencoders. (Variational) Autoencoders in the image domain have demonstrated an ability to encode semantic information into latent spaces [149]. However, applying them to graphs remains notably challenging, particularly for graphs featuring rich edge attributes and/or multigraphs [150]. A significant challenge lies in the graph generation or decoder component. Simonovsky et al. [150] have implemented an approximate graph matching and have applied it to Autoencoders. However, their approach is restricted to small graphs, with up to 38 nodes and singular connectivity between nodes. Other methods [151] expect the network to implicitly learn the correct matching. This approach remains viable but involves an enormous computational effort, considering the findings of Xu et al. [152].

Another approach involves constructing a data-driven encoder capable of discerning and quantifying differences between graphs. This encoder can also identify graph classes by recognizing clusters in the encoding space. This work implements and analyzes such an encoder, employing contrastive learning to facilitate the comparison of traffic scenes.

Zhu et al. [153] offer a comprehensive overview of diverse contrastive learning approaches on graphs, detailing design considerations such as data augmentation, contrasting modes, contrastive objectives, and negative mining strategies. Limited literature exists on the topic of contrastive learning for embedding entire graphs [154, 155, 156]. For instance, InfoGraph [156] discriminates representations of entire graphs from single nodes of other graphs without graph augmentation. Their objective is to capture shared aspects within substructures, maximizing mutual information between entire graphs and nodes. Another approach utilizing data augmentation is presented by

Hassani et al. [155]. They augment graphs using diffusion views but refrain from augmenting initial node features. Similar subsets of nodes from both graphs are sampled, akin to cropping in the visual domain. Additionally, two independent GNNs are employed, sharing MLPs for generating node and graph encodings. You et al. [154] investigate the impact of various augmentations, directly applied at the graph level, on contrastive graph learning. The studied augmentations include node dropping, edge perturbations, attribute masking, and subgraphs.

5.2 Methodology for Clustering Traffic Scene Graphs

The majority of related work on clustering and comparing traffic scenarios employs a similarity metric for time series. However, in this work, temporal information is not to be compared; rather, the similarity of scenes is to be described. Consequently, an alternative approach must be developed. The approach utilized by King et al. [146], for the description of maneuvers, is comparable to the heuristic on which the SSG is based. Nevertheless, their work continues to compare trajectories.

A graph contrastive approach is pursued to determine a measure of similarity between two traffic scenes. The basic idea of contrastive learning is that the representations of the same similar, *positive* samples in the latent state space are contracted, while *negative* samples are shifted away from the initial sample (*anchor*). In many application areas, often no label of the data is available. A positive sample can then be generated by automated data augmentation. Negative samples can be generated by randomly selecting from the remaining data. In recent years, contrastive learning, i.e. the combination of negative and positive samples, has been shown to improve the performance of self-supervised representation [157, 158, 159, 160]. Here, the work of Ma et al. [122] is also worth mentioning, which uses trajectory data of traffic participants to cluster traffic scenarios in a self-supervised manner.

In contrast to related work (see Section 5.1.2), this approach (see Figure 5.1) deviates from initiating the process with graphs as input. Instead, it starts with the generation of these graphs. The central phase involves direct augmentation of traffic scene data at the object list level ($S_{\mathbb{R}^2}$), resulting in augmentation within Cartesian space. Subsequently, a graph triplet is automatically generated. The objective of this methodology is to minimize the distance between the embeddings s_0, s_+ of two similar graphs, G_0 and G_+, in the embedding space S. In addition, it is important that the embedding s_- of a dissimilar graph G_- is as distant from s_0 as feasible. This is achieved by implementing a contrastive learning approach to address this graph representation problem. This approach ensures the generation of only valid scenes by incorporating the street layout into the augmentation process.

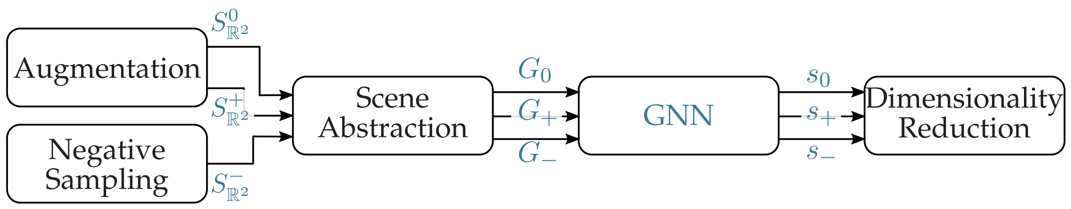


Figure 5.1: Pipeline for clustering traffic scenes

5.2.1 Augmentation and Graph Generation

The initial stage in the processing pipeline involves traffic scenes from the INTERACTION dataset (refer to Section 3.1). Within the employed methodology, the manipulation of the traffic scene takes place directly on the object list, in contrast to altering the graph features. This approach provides a notable advantage in preserving the generation of realistic and cohesive traffic scenes. In contrast, making random additions, deletions, or modifications of edges and nodes within the graph representation can lead to traffic scenes that are improbable or impossible in reality.

The positional and velocity attributes of individual traffic participants undergo random modifications. Traffic participants i within a scene, whose entity state is denoted by $\mathcal{X}^i$, are sampled uniformly with a probability of P_{entity} for the purpose of modification. The perturbation in the position (x, y) is modeled using normal distributions $N^x \sim (x, \sigma_{pos})$ and $N^y \sim (y, \sigma_{pos})$, while the variation in velocity $(\dot{x}, \dot{y})$ is characterized by $N^{\dot{x}} \sim (\dot{x}, \sigma_{vel})$ and $N^{\dot{y}} \sim (\dot{y}, \sigma_{vel})$.

This augmentation step is then repeated for the whole dataset to generate more positive samples. During augmentation, care is taken to ensure that no collisions occur due to the displacement of traffic participants. If two bounding boxes of two traffic participants overlap after augmentation, the scene is augmented again until there are no overlaps between traffic participants.

In Figure 5.2 the anchor $S^0_{\mathbb{R}^2}$ (Figure 5.2a) the positive sample $S^+_{\mathbb{R}^2}$ (Figure 5.2b) and a negative sample $S^-_{\mathbb{R}^2}$ (Figure 5.2c) are depicted. Figure 5.2d shows a magnified comparison between an original traffic scene (Figure 5.2a) and an

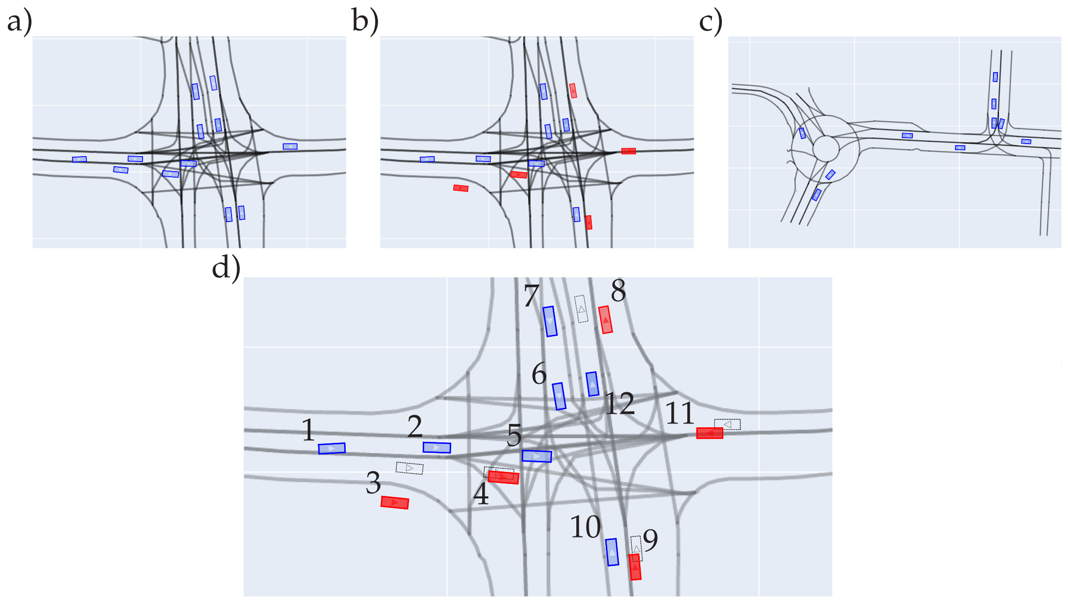


Figure 5.2: Scene triplet, consisting of the anchor (a), the augmented, positive sample (b) and a random, negative sample (c). Detailed comparison of the augmentation process (d). The position change of a traffic scene $S^0_{\mathbb{R}^2}$ (blue) and its augmented version $S^+_{\mathbb{R}^2}$ (red) of randomly selected traffic participants

augmented version (Figure 5.2b). Blue rectangles represent the initial poses of traffic participants, which were not modified. Red rectangles represent the augmented poses. For a better visual representation, the original poses are included as black dotted rectangles. While most positions undergo small alterations (see 4, 9, and 11), the random shift introduces the possibility of lane changes among traffic participants. For instance, vehicle 8 has been previously in the same lane as vehicle 12, and after the augmentation it is still in front of it, but in an adjacent lane. The augmentation process may move traffic participants beyond the boundaries of road geometry, as exemplified by the augmentation of vehicle 3. Consequently, this specific vehicle is omitted from subsequent analysis. Furthermore, any vehicles that have been shifted excessively into the opposing lane are excluded from the resultant scene graphs.

After the augmentation process, both the altered and the original traffic scenes undergo conversion into respective scene graphs denoted as G_+ and G_0. Both graphs differ mainly in their attributes on the edges and nodes. However, larger shifts (see vehicle 2 and 13) can also cause the graph topology to change or nodes to be removed. The traffic scene is converted from object lists into a graph using the SSG model (refer to Section 3.3). These scene graphs function as the input format for the subsequent machine learning processes.

In some cases, traffic scenes with few vehicles may not have edges in the graph representation. This can occur when two vehicles are driving on oncoming lanes that do not cross each other, or when one vehicle is driving alone on an intersection. Such graphs without edges are not considered in this framework, because the information propagation between nodes plays a crucial role in this approach. However, this is not a limitation of the presented approach, but explicitly filters trivial correlations.

In summary, the scenes are augmented specifically on the basis of a pattern and not purely randomly. The idea behind this augmentation is to create a positive sample based on the original scene, which should be different, but still similar enough that there are no fundamental changes in the traffic scene. The velocities and positions are reasonably adjusted based on the initial scene. By incorporating the scene graph model and the road layout, disproportionate modifications and the resulting structure of the graphs can be limited to realistic examples.

5.2.2 Model Architecture

To clarify the notation in this chapter, a node v^i_{scene} and its node attributes u^i are written as h^i_0. The model architecture used for encoding a traffic scene graph into an embedding is a GNN with two layers (MP_1, MP_2), a readout function R in form of a summation over all hidden node states and a projection head (ϕ), as seen in Figure 5.3. The nodes' neighborhood information is captured by two successive message passing operations MP_k. In each GNN layer $k \in \{1, .., K\}$, the state of the i-th node h^i_k is updated by firstly calculating a message along all incoming edges to the node. The message function m_k, which returns a message for all neighboring nodes $j \in \mathcal{N}(i)$ combines the features of the root node i and incoming nodes j, as well as the edge features e^{ji}_{attr} (see Section 3.4.2 and Section 3.4.3). Afterward, an aggregation function $\bigoplus$ is applied to reduce the arbitrary amount of messages to one of fixed dimensionality. This is the characteristic of GNN, because depending on the node, its neighborhood can vary in size, so the amount of information considered can vary. Therefore, it is crucial to choose a permutation invariant aggregation function. min, max, sum or mean operators are commonly used in this domain [161]. The resulting message is then passed through a final update function γ, alongside the original node attributes of the node's state h^{k-1}_i, resulting in the final new state h^k_i.

The message passing approach, which is done in each of the message passing blocks (MP_1, MP_2), can be described by the following formula:

$$h_k^i = \gamma_k(h_{k-1}^i, \bigoplus_{j \in \mathcal{N}(i)} (m_k(h_{k-1}^i, h_{k-1}^j, e_{attr}^{ji}))). \tag{5.1}$$

The resulting node states of all h_K are merged by a readout layer R. The resulting latent feature vector of the graph is then mapped onto the final feature space S by an MLP ϕ (projection head) with an intermediary activation function.

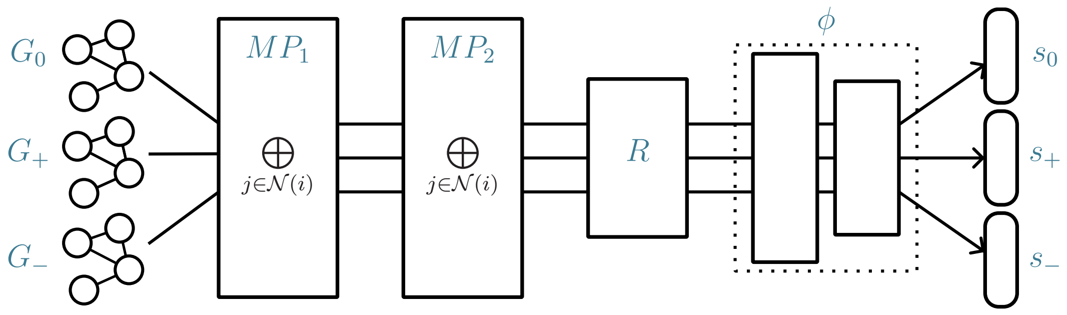

G_0
G_+
G_-
MP_1
$\bigoplus_{j \in \mathcal{N}(i)}$
MP_2
$\bigoplus_{j \in \mathcal{N}(i)}$
R
ϕ
s_0
s_+
s_-

Figure 5.3: Graph encoding model mapping graphs to the embedding space. Each graph of the triplet (G_+, G_0, G_-) is fed through the model individually. (Adapted from [9]).

5.3 Experiments

Two slightly different architectures with different training data are used for training and subsequent evaluation of the method. Experiment 1 is intended to confirm the basic functionality of the method and is therefore performed on a very limited dataset. It uses 2842 different traffic scenes originating from one location (Roundabout OF) in the dataset. In the second experiment, however, different road geometries from different locations are examined. Table 5.1 shows the locations and the number of anchor traffic scenes used. However, the 9306 traffic scenes are only the real samples, the augmented traffic scenes used in the training process are added.

Road Type	Country	Name	Num. Scenes
Roundabout	GER	OF	1822
Merging	GER	MT	863
Intersection	USA	EP0	1651
Intersection	USA	EP1	1349
Roundabout	USA	EP	1769
Roundabout	USA	SR	1852
Total			**9306**

Table 5.1: The composition of the utilized dataset, which was derived from the INTERACTION dataset.

5.3.1 Training Setup

Each triplet of a traffic scene, consisting of the anchor G_0, the positive graph sample G_+ and the negative graph sample G_- are mapped onto the embedding space S.

Learning is conducted using the triplet loss function $\mathcal{L}_{triplet}$ from Equation (5.2).

$$\mathcal{L}_{triplet}(s_0, s_+, s_-) = max\big(d(s_0, s_+) - d(s_0, s_-) + M, 0\big), \tag{5.2}$$

with s_0, s_+ and s_- being graph encodings ($s_0, s_+, s_- \in S$). s_0 is the encoding of the original graph, the so-called anchor. The augmented encoding, the positive sample, is s_+ and a sampled, thus differing graph is used as the negative example s_-. Sampling negatives is a broad research area itself [153]. Negative samples are sampled at uniform from the rest of the batch. The function d can be any suitable distance metric, like Euclidean distance or cosine similarity. In this application, the best results are achieved with the Euclidean distance (Equation (5.3)). The margin M is a value that may be freely configured; an increase in the selected value results in a greater separation of negative samples in the embedding space S.

$$d(s_0, s_+) = |s_+ - s_0|_{\mathcal{D}} \tag{5.3}$$

The MP_1 layer incorporates both node and edge features, whereas MP_2 exclusively operates on hidden node states. In the latter case, values assigned to the edge attributes e^{ji}_{attr} have been set to 0. This is because the edge information is already contained in the first message passing operation inherent in the node information for the second layer MP_2. In Experiment 1, a single layer is employed for both the message function m and the update function

γ. Leaky Rectified Linear Unit (Leaky ReLU) activation functions are applied to introduce nonlinearity, and the hidden node states are represented in a 60-dimensional space. The final output of the projection head is processed through a tanh function. Additionally, the embedding dimensionality $\mathcal{D}$ is set to 2 (S_0, S_1). The selection of this minimalistic design aims to evaluate the general functionality of the architecture. Moreover, S is configured to facilitate visual clustering without resorting to dimensionality reduction to two or three dimensions.

Experiment 2 deviates from single linear layers by incorporating two-layer MLP in both the m and γ of the message passing layers. The dimensionality of the embedding space $\mathcal{D}$ is increased to 12 in this experiment to enable a more diverse range of descriptions. Regularization is implemented through dropout in the message passing layers, and the projection head is constructed as a three-layer MLP.

As readout function R, the mean operation is used over all node features of a graph. As an aggregation function $\bigoplus$ the sum operation is used. Negative sampling is conducted uniformly over the batch, excluding the positive sample. The training conditions can be seen in Table 5.2.

Parameter	Value
Learning rate	0.001
Margin M	0.5
Embedding dimensionality $\mathcal{D}$	2/12
batch size	400
Epochs	400
Optimizer	ADAM

Table 5.2: Training hyperparameters

The aforementioned triplet loss $\mathcal{L}_{triplet}$ in Equation (5.2) is used in conjunction with the Adaptive Moment Estimation (ADAM) optimizer. A learning rate of 0.001 is used, and the triplet margin M is set to 0.5. The distance function d used in the triplet loss is the Euclidean distance. Negative sampling was conducted uniformly over the batch, excluding the positive sample.

For generation of the augmented data $\sigma_{pos} = 1.5\,\mathrm{m}$ and $\sigma_{vel} = 2.0\,\frac{\mathrm{m}}{\mathrm{s}}$ with $P_{entity} = 0.5$ is used (see Section 5.2.1). The training parameters, aggregation functions and all other hyperparameters are determined through a systematic and empirical process, employing a grid search approach.

The dataset is first evenly split into a holdout set used for clustering and evaluating the results (20%) and a train/test (80%) set. The train/test set is then again split into train (80%) and validation (20%) for parameter tuning. The graphs used have both node and edge features. For node features, the object classification and its velocity are used. The directional information of vehicle velocities or orientations are excluded in this approach, as the aim is to generalize traffic scenes without relying on their absolute position (see Section 3.3). The emphasis is on the interdependent relationships between the different traffic participants, rather than their absolute, individual characteristics.

5.3.2 In-Depth Analysis

One major challenge with this analysis problem is the lack of ground truth data or labels for comparison. Therefore, both quantitative and qualitative evaluations are conducted within the embedding space to determine its meaningfulness and behavior with respect to parameter changes. Additionally, the generated clusters within the embedding space are qualitatively examined to observe differences in similar embeddings within the real traffic scene.

Embedding Space

Examining the encoded traffic scenes of Experiment 1 through a scatter plot (Figure 5.4) reveals distinct clusters. Employing statistical information derived from the original graphs to colorize the encodings yields some perspectives on these clusters. Representing the number of vehicles in each scene in a similar manner exposes a gradient, as illustrated in Figure 5.4a, indicating that scenes with a higher number of cars tend to gather in the lower left, while those with fewer cars tend to aggregate in the upper right. This observation underscores the pivotal role of vehicle count as a determinant for the network's discrimination of traffic scenes. This phenomenon is also evident in Figure 5.7a, emphasizing that, for instance, a scenario depicting a traffic jam necessitates the presence of multiple vehicles. Consequently, it is evident that a discernible structure manifests on a global scale, indicating the network's ability to capture meaningful patterns.

The impact of modifying singular attributes in the original graph on the encoding is visually demonstrated in this study. To enhance clarity, a set of 25 random traffic scenes, situated in distinct clusters, is chosen. The velocity of all traffic participants is systematically increased by $0.5\frac{\mathrm{m}}{\mathrm{s}}$ in ten increments, resulting in eleven diverse scenes for each instance. Both the original samples and their corresponding modified versions are encoded

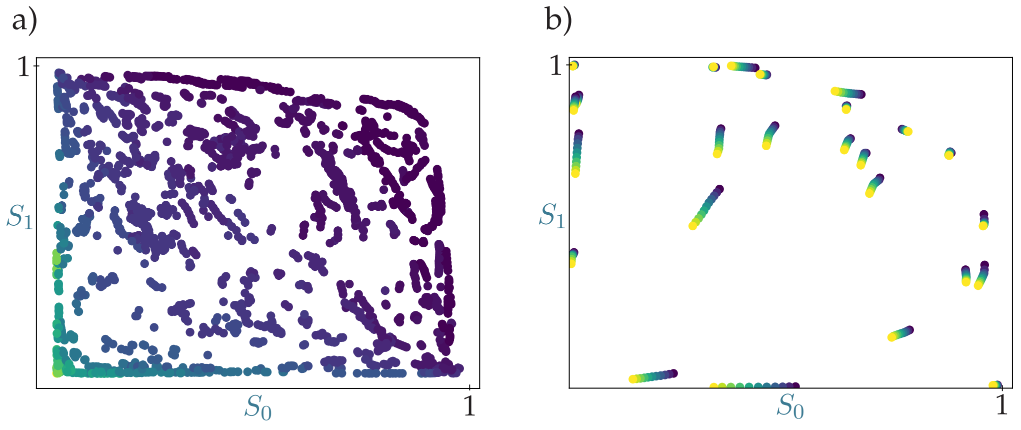


Figure 5.4: Encoded traffic scenes, mapped to S, colored according to the number of vehicles (2 ... 8) (a). Exemplary encoded traffic scenes, mapped to S, with increased velocities (0 m/s ... 5.5 m/s). Violet dots represent the initial scene (b). (Adapted from [9])

within S and are presented in Figure 5.4b. A noticeable trend emerges as many samples exhibit a tendency to move towards the lower left as their speed increases. The extent to which an altered data point is displaced serves as an indicator of the dissimilarity between the resulting traffic scene and the original configuration. This observation is particularly pronounced in traffic scenarios featuring multiple participants driving closely in tandem, as depicted in Figure 5.4b bottom left and Figure 5.7a.

Experiment 2 also analyzes the distribution of information within the embedding space. Of particular interest is the continuity within the embedding space. To effectively visualize this high-dimensional space, dimensionality reduction techniques are employed, specifically PCA, which reduces $\mathcal{D}$ to two dimensions (first two components). Subsequently, colors are assigned to the samples based on certain handcrafted features.

An exemplary feature analysis is presented in Figure 5.5a, where the coloring is determined by the average speed of cars ($mean(\vec{v})$). Notably, a gradient is observed from the top-left to bottom-right, representing a transition from high to low speed. This suggests the preservation of scene-describing continuity within the embedding space.

Another significant feature analysis is showcased in Figure 5.5b, where the coloring is based on the number of cars $\|V_{car}\|$ present in the scene. Particularly on the left-hand side of the figure, six distinct line-shaped clusters are evident. Similar to the previous example, a gradient is observable, this time from left to right. Remarkably, these lines encode the number of cars in an ordinal fashion, with the leftmost line predominantly comprising scenes with

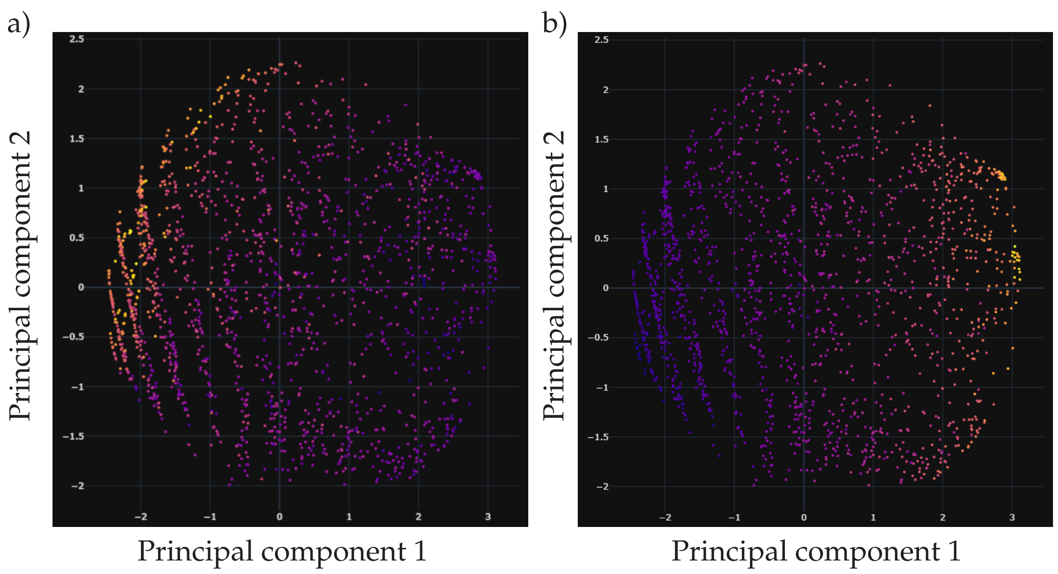


Figure 5.5: The PCA-reduced embedding space (two most significant dimensions) colored according to the average velocity 0 m/s ...11 m/s of the cars present in the scene (a) and according to the number of vehicles 1 ...22 (b). (Adapted from [4])

one car, and each subsequent line incorporating one additional car. After the sixth line, this pattern appears to dissipate, potentially due to an increased number of cars in scenes leading to more intricate interactions. Distinguishing between such scenes likely requires a more sophisticated approach by the graph representation model. Further analysis of additional features (see Table 5.4) reveals similar observations.

After an examination of the structure of the embedding space S and the distribution of data points, a quantitative analysis of the general representability of the model is presented in the subsequent discussion. The primary objective is to ascertain the effectiveness of the trained model in embedding similar traffic scenes within the aforementioned embedding space. To evaluate this, experiments are conducted on a holdout set, utilizing data that has not been encountered during the training phase. The assessment of the network's discriminative capability is determined by the proximity of positive samples s_+ to the anchor s_0 in relation to negative samples s_-, thereby ensuring accurate classification. This assumption is formally expressed in Equation (5.4).

$$d(s_0, s_+) < d(s_0, s_-) \tag{5.4}$$

The accuracy of the trained model is detailed for various partial datasets in Table 5.3. Here, $\bar{d}$ denotes the average distance measured for each location. The overall accuracy of the model on the test dataset reaches approximately

Street Type	Location	Accuracy	$\bar{d}(s_0, s_+)$	$\bar{d}(s_0, s_-)$
Roundabout	OF	0.989	0.385	2.921
Merging	MT	1.0	0.471	3.079
Intersection	EP0	0.99	0.439	2.628
Intersection	EP1	0.993	0.458	3.057
Roundabout	EP	0.994	0.466	2.944
Roundabout	SR	1.0	0.439	3.108
Total		**0.993**	**0.439**	**2.927**

Table 5.3: Classification accuracy and average distances between anchor, positive and negative sample

99.3%. Notably, for locations MT and SR, all samples within the holdout set are correctly classified. The consistent distances observed between anchor, positive, and negative samples provide evidence that the model has effectively generalized its discriminative properties.

The task of labelling traffic scenes presents a challenging problem in the field of research. While it may be feasible to establish a concise set of rule-based labels, such as "following" or "overtaking", for scenarios involving only a small number of vehicles, the complexity of traffic scenes far surpasses this scale, often comprising several dozen vehicles at maximum. Attempting to manually label such complex interactions or relying on rule-based methods would therefore prove unproductive. As a result, an alternative approach becomes imperative for the purpose of validation. This approach utilizes features derived from the graph structure itself (see Table 5.4). These features, although not as discerning as categorical labels, offer valuable aggregated information regarding the defining characteristics of a given traffic scene. Those features are used to evaluate how well the embedding space captures semantic information of the traffic scenes. Again, the holdout set is employed to regress a number of handcrafted features via a small four-layer MLP (30 hidden dimensions) utilizing the embedding space S. For this, the holdout set is split again into a train/test (80/20). The results of the regression errors for the test set is shown in Table 5.4. Training was conducted over 2500 Epochs, with a learning rate of 0.001, 10% dropout, optimized using ADAM. The model was trained on Mean Squared Error (MSE). To bring the resulting errors into context, the Mean Absolute Error (MAE) of the prediction in combination with the Mean, Standard Deviation (Std), Min and Max of the regressed feature in the test dataset is also included.

By evaluating the absolute error in relation to the usual ranges of the corresponding regressed variables, the considerable significance of the embed-

Regression Variable	MSE	MAE	Mean	Std	Min	Max
$\|\|E_{lon}\|\|$	0.572	0.553	1.937	1.37	0.0	8.409
$\|\|E_{lat}\|\|$	0.885	0.218	0.243	0.388	0.0	2.067
$\|\|E_{int}\|\|$	0.367	0.588	3.195	2.097	0.0	12.85
$\|\|E\|\|$	0.108	3.490	58.45	53.611	1.0	398.0
$\|\|V_{car}\|\|$	0.289	0.772	8.160	3.75	1.0	22.0
$mean(\nu)$	0.434	0.635	4.349	1.863	0.0	10.894

Table 5.4: Regression performance of the embedding space onto handcrafted features.

dings can be established. For example, the number of cars $||V_{car}||$ can be estimated with a deviation of less than one car on average. Similar results can be observed for the number of edges, velocities of cars and the number of interaction types ($||E_{lon}||$, $||E_{lat}||$, $||E_{int}||$), that are normalized to the number of vehicles in the regarding scene.

To conclude the investigation of the embedding space, three important properties of the approach are tested and observed. First, it is shown that similar graphs are closer to each other compared to dissimilar graphs. Second, it can be established that the fixed-dimensional embedding space contains important information that can be effectively utilized in subsequent tasks. Finally, it is confirmed that the embedding space has continuity for at least some of the traffic scene determining information.

Clustering Evaluation

To conduct a thorough examination of the formed clusters in Experiment 1 with respect to their semantic meaning, the DBSCAN clustering algorithm ($\epsilon = 0.05$, $samples_{\text{min}} = 5$) is applied to the encodings. The selection of parameters is determined through the utilization of the elbow method [162]. This yields a total of 75 distinct clusters and outliers (depicted in black), as illustrated in Figure 5.6. To improve clarity, fewer colors are used to depict the clusters; thus, only points of clusters that have the same color and are in close spatial proximity are connected. Points of the same color that are not close together belong to different clusters.

For the qualitative assessment of the method, two traffic scenes from each of two distinct clusters are presented in Figure 5.7. Broadly speaking, each cluster encompasses traffic scenes that unfold consecutively over time, aligning with the initial expectations. This observation indicates, that nearly identical scenes are grouped together. Furthermore, scenes recorded at different times yet exhibiting a similar vehicle arrangement are also allocated to the same cluster. For instance, in Figure 5.7a, several vehicles can be observed traveling in a convoy. Despite variations in specific areas, the commonality of a vehicle chain is consistently represented in every sample within this cluster.

This pattern is prevalent across the majority of clusters. Notably, the case exemplified in Figure 5.7b is noteworthy, where the network autonomously groups traffic scenes involving a parked vehicle (red circle). Despite the inclusion of velocity information for each participant, the classification "parking" is not considered.

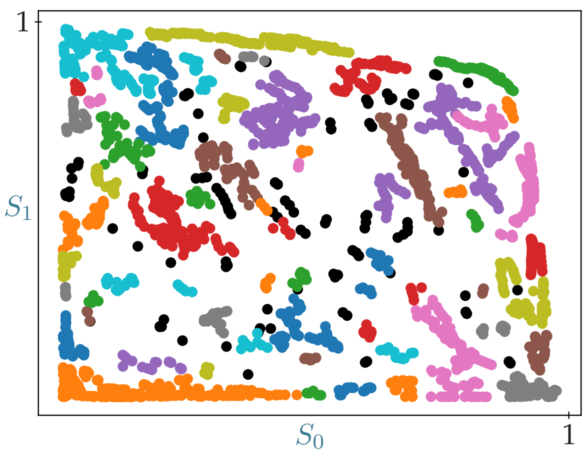


Figure 5.6: Each dot represents a graphic scene embedding in the two-dimensional S. The samples within a given cluster are identified by the same color and are located in close spatial proximity to one another [9].

To assess the encodings on a deeper semantic level in Experiment 2, clustering of the test dataset is applied. Initially, dimensionality reduction is executed using the UMAP algorithm [163] ($n_{\text{neighbors}} = 5, min_{\text{dist}} = 0.0, n_{\text{components}} = 2$). The resulting two-dimensional space undergoes agglomerative hierarchical clustering (refer to Figure 5.9). The determination of the number of clusters involves testing a reasonable range ($[2..500]$) against their silhouette score, and the version with the highest score is selected. It is observed that the silhouette score generally increases with the number of clusters until approximately 250 clusters, reaching a score of around 0.75. Beyond this point, the silhouette score starts to decrease. To facilitate manual evaluation of the clusters, the number of clusters is capped at 32, with a silhouette score of 0.687.

Figure 5.8 presents six traffic scenes selected from two distinct clusters of Figure 5.9. The top three scenes from the first cluster (a) depict a small group

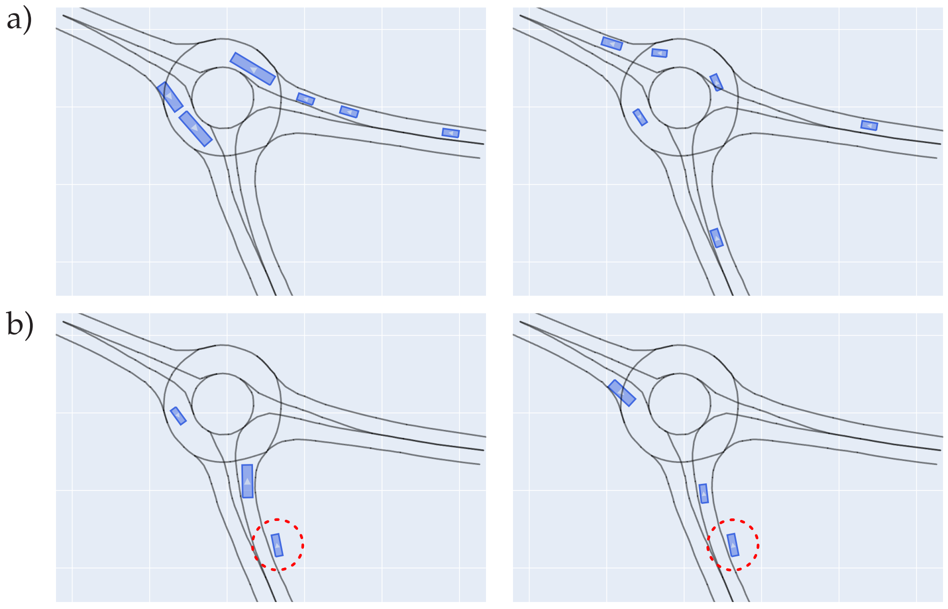


Figure 5.7: Two exemplary traffic scenes of two clusters. Vehicles are described by blue rectangles, road boundaries are indicated with black and white lines. The first cluster has the characteristic that several vehicles drive behind each other (a). The second cluster several vehicles are in the roundabout additionally there exist a parking vehicle (red circle) (b). (Adapted from [9])

of two traffic participants driving behind each other, individual participants in an intersecting lane, and one vehicle that does not interact with any other vehicles. It is noteworthy that the approach not only captures similar scenes for the same road type but transcends road types to make scenes on different roads comparable. The traffic scenes in the second cluster (Figure 5.8b) showcase long queues of traffic participants. Consistent with the aforementioned clusters, the remaining generated clusters exhibit analogous patterns within their traffic constellations. In general, clusters portraying similar traffic scenarios are discovered across diverse road configurations in various locations.

This qualitative analysis demonstrates that the model has successfully acquired a semantic representation of traffic scenes. However, in both Experiment 1 and Experiment 2, certain traffic scenes have been grouped into a cluster where no identifiable pattern is present upon inspecting the original scene. It is evident from the presented data that an automated algorithm does not consistently distinguish between traffic scenes and traffic situations based on the same characteristics as humans, a phenomenon that was already predicted by Hauer et al. [51] in the context of automatic clustering

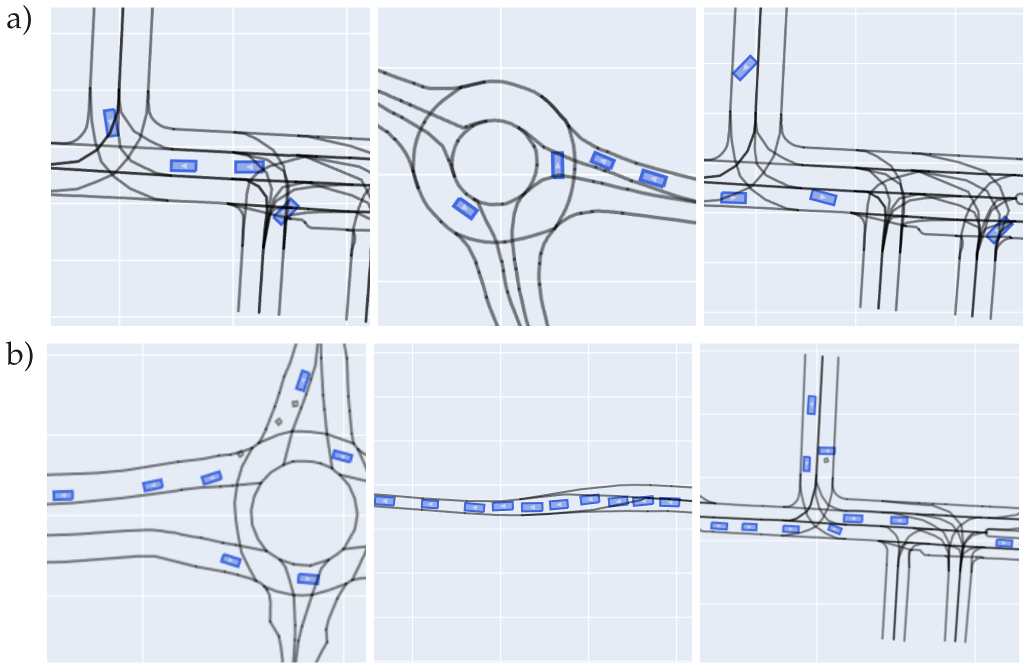


Figure 5.8: Three examples of traffic scenes, each of two different clusters (a, b) from Figure 5.9 (Adapted from [4])

of traffic scenarios. Therefore, these clusters cannot be easily classified by humans. However, there are shared similarities within these clusters that may be justified on their own, but are difficult to classify based on human common sense.

5.4 Summary

With the self-supervised graph clustering approach presented above, the questions in Research goal 3 are addressed. It presents the effectiveness of GNN when coupled with contrastive learning as a viable approach to encode complex traffic scenes. The methodology facilitates scene comparison without relying on hand-crafted labels and features, instead leveraging the inherent interaction graph structure between vehicles in an automated manner. The resulting embedding space demonstrates the ability to distinguish between positive and negative samples from an anchor, while also capturing essential graph-level properties. A qualitative analysis of formed clusters reveals the approach's capability to successfully identify similar traffic scenes on different street types.

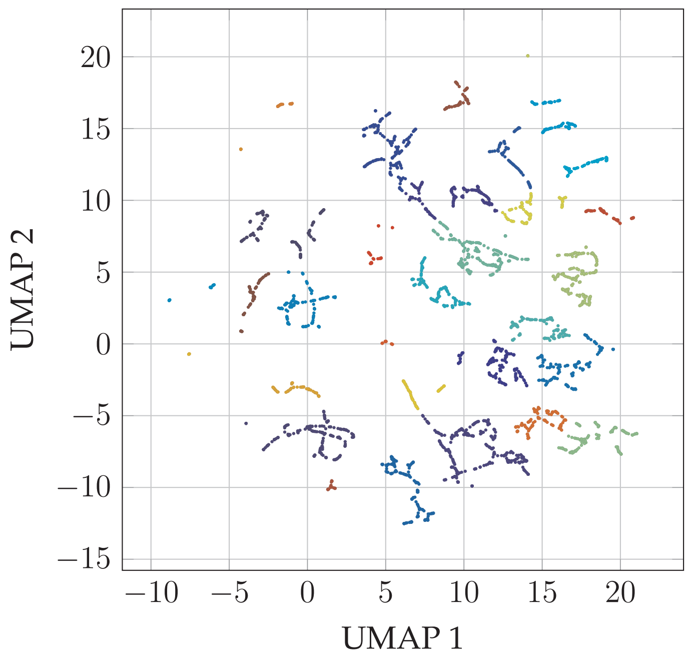


Figure 5.9: The embedding space S is dimensionally reduced to a two-dimensional space using the UMAP algorithm. Hierarchical clustering is employed to assign data points to 32 distinct clusters, which are represented by different colors. (Adapted from [4])

6 Driving Behavior Models

The goal of a driver model is to create an intelligent, knowledge- and rule-based system that reacts according to the given environment in order to create a trajectory. Research into motion and trajectory prediction in road traffic has been driven in particular by the ongoing development of highly automated driving. Several influential factors significantly contribute to ensuring safe navigation from one point to another. While some factors, such as road infrastructure, are inherently static, the behavior of various traffic participants undergoes dynamic changes over time. Therefore, the adaptive response of an autonomous vehicle, including actions such as steering or braking, heavily depends on a comprehensive understanding of the changing behavior of nearby traffic participants. The variation in responses among traffic participants, depending on their specific situational contexts, highlights the need for a model that predicts their intentions. The creation of such a model requires an understanding of not only the exact positions of traffic participants but also their interrelationships. Human drivers, for example, gain insights about the underlying traffic dynamics directly from their perceptual environment. This understanding is augmented by their knowledge of traffic regulations and observed behavioral patterns.

This chapter addresses the behavior models that will be incorporated into the simulation in the subsequent chapter (see Chapter 7) for the initialization of the respective actors in a scene. The objective is to develop behavior models that reflect a broad spectrum of driving behaviors observed in real-world scenarios. For this reason, machine-learned driving behavior models based on the SSG (see Chapter 3) are discussed on the one hand. On the other hand, classic, rule-based models are also discussed.

Parts of this chapter have been published previously in the following peer-reviewed publications [7, 5], and some of the content has been adopted verbatim.

6.1 Related Work

The modeling of driver models for vehicles has been a long-studied topic. The aforementioned models can be classified into two principal categories. The first category comprises models designed for the purpose of controlling a vehicle and calculating its behavior for a specific driving task. In this context, the models make use of explicit measured variables and specified behavioral parameters. The second category includes models designed for the purpose of prediction. Various physic-based, mathematical, and machine-learned models are used to predict the behavior of other traffic participants. The boundary between the models is not clearly defined, and most of them can be replaced by a model from another category.

In the first category, Gartner et al. divide models into microscopic, mesoscopic, and macroscopic traffic models [164]. The mesoscopic level views traffic as a fluid, enabling simulation of road network occupancy. At the microscopic level, vehicles are individually stored in more complex models, allowing for more specific mapping of driving tasks [165]. These levels, which primarily calculate vehicle groups or vehicle flows, are not relevant for this work since it is not possible to consider individual vehicles. In contrast, microscopic models consider vehicles at the highest level of complexity and take into account the respective environment for each vehicle individually.

Classic vehicle-following models operate on the same principle: they use mathematical rules to model vehicle behavior in a deterministic manner. The model's reaction, typically acceleration, is determined by an external input. One commonly used model for calculating a vehicle's acceleration is the work of Gazis et al. [166]. This model considers the speed of the vehicle and the distance from the vehicle in front, with two model constants, and includes sensitivity to the distance between vehicles. Treiber et al. [167] have developed the Intelligent Driver Model (IDM) based on this basic idea. The model incorporates empirically measured behavior of traffic participants, resulting in a more concrete description of their behavior. Additionally, the model parameters can be interpreted and, like the model of Gazis et al., it allows for fast numerical calculation of acceleration. Another driver model that uses similar model parameters as the models above, but introduces reaction time and a disturbance variable is presented in the work of Krauß [168] obtaining deviations from optimal driving. This model is utilized in the SUMO traffic flow simulation [169]. For an in-depth analysis of driver models, see the work of Gartner et al. [164].

Detecting hazardous scenarios and responding appropriately to prevent or minimize accidents is a significant challenge. This includes predicting traffic states and assessing the potential level of hazard in the upcoming situation.

Prediction of other traffic participants is especially necessary in the case of intelligent vehicles.

Lefèvre et al. [170] categorize prediction models for traffic participants into three subgroups: physics-based, maneuver-based, and interaction-based models. Physics-based models utilize kinematic dynamics models to calculate the behavior of traffic participants, taking into account factors such as acceleration, position, and heading. These models are efficient and widely used to compute criticality metrics and tracking algorithms. However, forecasting capabilities of these models are typically very limited, especially for longer forecasting horizons. Maneuver-based models predict the driver's intended driving behavior and future plans, such as turning or overtaking. The predicted trajectory is then determined, in some cases, using physics-based models. Interaction-aware models expand on the concept of maneuver recognition by considering the influence of other traffic participants on a given participant's behavior. For a review of these models using traditional methods, please see Lefèvre's survey paper [170].

The current state of the art of trajectory predictors uses machine learning techniques. Most models either fall into the maneuver prediction category or, in particular, newer models fall into the interaction-aware model category. These recent developments have seen the use of neural networks, particularly Recurrent Neural Networks (RNNs) based on Gated Recurrent Networks (GRUs) or Long Short-term Memorys (LSTMs) [171]. Current methods for machine learning are primarily based on supervised learning. However, they often use complex energy functions as loss functions or involve multiple loss functions, which must be determined manually.

Architectures can be broadly classified into two categories: one for processing ordered data, mainly synthetic images, and those for non-ordered, graph data. CNNs are commonly employed to incorporate image-encoded contextual information, whereas RNNs or transformers are generally applied to non-image data. Conversely, graph-based models represent vehicles or road segments as nodes, employing graph convolution to share dynamic information across multiple steps within their graphs.

6.1.1 Image-based Trajectory Prediction

In the field of image-based trajectory prediction, Bansal et al. [90] utilize a simple CNN to extract features from an abstract birds-eye view (top-down) image representation. These features are then processed in an RNN to predict the movements of an agent. Parallel RNN streams provide distinct loss information using both vehicle and contextual data. Hong et al. [172] use a similar approach where road geometry, traffic light information and other traffic participants are combined into different tensors similar to a birds-eye view. The first two dimensions represent spatial positions in the top-down image. The third dimension contains content information, such as previous states or other feature information. The traffic participants have dynamic attributes such as speed and acceleration. The approach of Schörner et al. [173] is based on a similar, stacked image environment representation. In this representation, the subsequent states at a given point in time in the future are estimated. The prediction of a trajectory is achieved by iterative prediction.

MultiPath [174] utilizes state-sequence anchors that correspond to a model of trajectory distributions. During inference, the model predicts a discrete distribution over the anchors and regresses offsets from anchor waypoints along with their uncertainties. This results in a Gaussian mixture at each timestep. Chen et al. [91] propose a simplified approach to designing a vehicle controller, using a pure feed-forward CNN model, specifically the VGG16-Net [175], originally developed for computer vision classification and detection. In this non-recurrent model, the temporal dimension is achieved by transparently blending the graphical representation of the previous timesteps with the representation of the current timestep. Zeng et al. [92] postulate a CNN-based model to represent probability distributions of traffic participant's movements from vehicle sensor data for safe vehicle movement planning. The work of Cui et al. [176] represents one of the earliest applications that use multimodal trajectory prediction and introduce the multi trajectory prediction loss to overcome the mode collapse problem.

6.1.2 Graph-based Motion Prediction

Recently, the potential of graph representations for trajectory prediction has been recognized. They have the advantage over image-based approaches, for instance, that information and especially its relationships can be represented explicitly. As a result, graph-based prediction architectures are at the top of the ranking lists of publicly available datasets [177, 178].

Diehl et al. [121] have created a graph representation of a highway scene connecting up to eight vehicles. They compare the trajectory prediction performance of Graph Convolutional Network (GCN) [179] with Graph Attention Network (GAT) [180] and propose modifications to the task of vehicle behavior prediction. Mo et al. [181] present an approach employing a GNN-RNN combination, drawing inspiration from Diehl et al. trajectory prediction method. In this approach, vehicles are represented by an RNN, and their interactions are modeled as a directed graph using a GNN. The model's output is subsequently subjected to further processing using an LSTM [182]. Li et al. [118] present a graph-based approach for interaction-aware trajectory prediction (GRIP), which focuses on highway scenes. The approach represents close vehicles as neighboring nodes in a graph. The authors of [183] propose a spatio-temporal graph dual-attention network for multi-agent prediction and tracking. It incorporates relational inductive biases, a kinematic constraint layer and leverages both trajectory and scene context information.

Gao et al. [120] present VectorNet, a vectorized definition of the scene where unified representations are learned from their vectorized form. The graphic extent of a road feature can be a point, a polygon, or a curve in geographic coordinates. A GNN is then used to incorporate the set of vectors, where each vector is treated as a node in the graph. This work is particularly relevant in the context of road representation, as it is used as a basis for many other works. Zhao et al. [184] present the Target-driveN Trajectory (TNT) framework, whose backbone is based on VectorNet. This approach consists of three steps: predict the agent's potential target states into the future, generate trajectory state sequences conditioned on targets, estimate trajectory likelihoods and final compact trajectory predictions. The work of Grimm et al. [185] proposes a highly heterogeneous graph structure to represent interaction between traffic participants and the static environment. The road layer representation is influenced by the VectorNet idea. The SEPT framework by Lan et al. [186], inspired by the well-known large language models, is also based on the principle of self-supervised learning. Using a transformer architecture, the trajectories and the road network are learned separately. The input representation of traffic participants is a tensor of the nearest traffic participants, including relative position and type for a given history. The road information is also based on the VectorNet approach.

The GNN-based approach presented in work of Ma et al. [122] utilizes the Frenet representation to enhance trajectory prediction in multi-agent interactions. In contrast to most prior work, edges are not set with all nearby traffic participants, but with specific traffic participants based on the interaction type. The attributes of the nodes and edges are generated using both a feature vector (velocity and relative distances) and a top-down image.

6.2 Machine Learning Models

Imitation learning models offer the possibility of replicating the real behavior of different traffic participants depending on the situation. Graph-based methods, which enable the mapping of explicit relationships between traffic participants, have particularly good prediction properties. The section introduces two architectures for traffic participant prediction (Relation-based Model and Hybrid Model) that focus on the interaction between individual traffic participants. The models based on imitation learning presented here employ the SSG described in Chapter 3 to predict a movement based on it. The Relation-based model predicts a longitudinal acceleration of the ego-vehicle based on the SSG. The second model, the Hybrid Model, additionally incorporates a top-down representation of the environment as an image into the model architecture, predicting a complete trajectory.

6.2.1 Architectures

Relation-based Model

This model is intended to capture the behavior of a traffic participant based on its immediate dynamic environment and to estimate the longitudinal dynamic behavior based on this information. The contextual information is exclusively provided by the SSG. This means that all traffic participants are represented by nodes in the graph and mutual relationships are represented by edges based on the relative position of the traffic participants with respect to the topology of the road network. No explicit road geometry information is included in the input representation. This prevents a concrete prediction of the steering angle. Accordingly, only the acceleration behavior $\tilde{a}$ is predicted.

$$\tilde{a}^i(t) = \frac{\nu^i(t+\Delta t) - \nu^i(t)}{\Delta t} \tag{6.1}$$

In the following experiments, Δt is set to 1 s, which approximately corresponds to the average acceleration $\tilde{a}$ over the next 10 timesteps.

The representation of the participant's environment is learned by using a GNN approach, where each participant is represented by a node. The nodes are updated based on the outgoing edges and their corresponding neighbors. The message passing approach employed here is based on the work of Gilmer et al. [187] (see Equation (6.2)).

$$h_k^i = h_{k-1}^i \cdot \Theta + \sum_{j \in \mathcal{N}(i)} \left(h_{k-1}^j \cdot \phi_{edge}(e_{attr}^{ij}) \right) \tag{6.2}$$

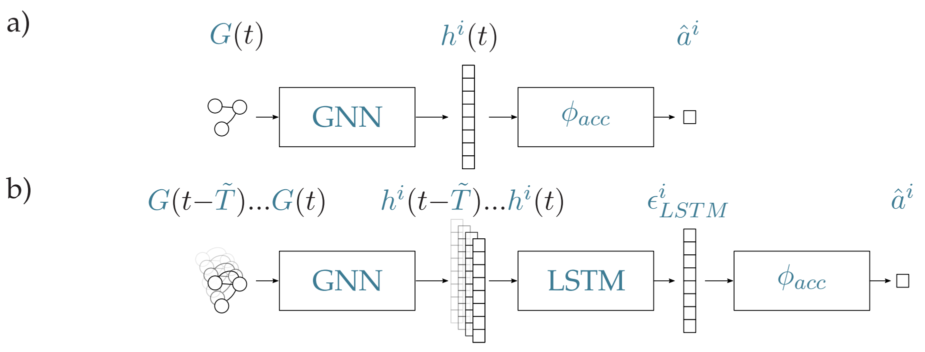


Figure 6.1: Relation-based trajectory prediction architecture for an ego-vehicle i, for the *Single Step* model (a) and the *Recurrent* model (b).

The message function comprises a learned MLP ϕ_{edge} that maps edge attribute vector to a weight matrix. The weight matrix is then multiplied by the attributes of the neighboring nodes h^j_{k-1}. The update function sums the message functions of all neighbors ($j \in \mathcal{N}(i)$) and adds them to the current state of the root node h^i_{k-1}, which is mapped by a learned weight matrix Θ. This results in the updated root node h^i_k.

Typically, GNNs are designed that the message passing step $k \in K$ can be performed several times in a row to propagate the information further in the graph. In this architecture, the number of message passing steps is limited to $K = 1$. As a result, the root node only receives the information of the neighboring traffic participants that are assigned a direct relation by the SSG. This is intended to show that the information content of the SSG is sufficient for the prediction and thus indirectly for the representation of a traffic scene. After the message passing step, each node in the graph contains a hidden state h^i_k, which is dependent on its neighborhood and the associated relations. Finally, h^i_K is mapped by ϕ_{acc} to a floating point number that describes the predicted acceleration $\hat{a}^i$.

In the applied architecture ϕ_{edge} has a hidden size of 32. Hidden states of nodes h_k have the size of 64. ϕ_{acc} is also an MLP with a hidden size of 128. The sizes of the hidden layers and all other hyperparameters are determined empirically with a grid search. All MLPs have an interposed Leaky ReLU as an activation function. In the following and in the experiment sections, this architecture is referred to as *Single Step* (see Figure 6.1a).

In addition to considering a single traffic scene as input, the influence of a temporal sequence of scenes on motion prediction is investigated. To capture the temporal information, an LSTM [182] architecture is applied. The idea of capturing temporal information of a graph is based on the works of Taheri et al. [188]. For each graph for t of the temporal sequence of the length $\tilde{T}$, the node information is propagated through the graph with the *Single Step* architecture (see Equation (6.2)). The state of each hidden node is then fed into an LSTM block, updating the LSTM encoding. This process is repeated for each graph in the temporal sequence, updating the LSTM encoding each time. The final LSTM encoding ϵ_{LSTM} of the *Recurrent* architecture (see Figure 6.1b), is then converted to an acceleration value by an MLP ϕ_{acc}, similar to the *Single step* architecture.

Hybrid Model

As shown in the state of the art (Section 6.1), two input representations are particularly common in trajectory estimation. The Hybrid Model architecture utilizes two separate machine learning methodologies based on two distinct representations of the environment. One method relies on a top-down image-based representation that incorporates spatial information, in particular road information, while the other uses a graph representation of the environment that is specifically designed to represent social contexts within a traffic scene. Both the graph representation of the Hybrid Model and the relational trajectory model are based on the SSG.

The top-down images of the grid-based environment representation are based on the work of Bansal et al. [90] and Chen et al. [91]. In contrast to the Relation-based trajectory model which is a joint prediction model, the Hybrid model predicts only one specific ego-vehicle for each prediction step (independent prediction). This means that if n vehicles are to be predicted in a scene, n prediction runs must be started. A corresponding view of the traffic scene is generated (refer to Figure 6.2), centered on the ego-vehicle. The ego-vehicle is colored in red, and its front is always oriented to the right in the local image coordinate system. Other traffic participants are shown as green rectangles, the length and width of the traffic participant is determining the size of the rectangle. Lane markings are represented by white lines. In addition to the current position of each traffic participant, their movement history is also represented by fading rectangles. The degree of fading corresponds to the time interval to the current state. The movement history is divided into ten states over a period of one second. Each square in the top-down scene represents an area of $53\,\mathrm{m} \times 53\,\mathrm{m}$ in reality. This size has been chosen to ensure that all traffic participants relevant to the ego-vehicle are within the field of view, particularly in inner-city traffic situations. The top-down view is then rasterized in $224\,\mathrm{px} \times 224\,\mathrm{px}$, whereby a pixel reflects approximately $0.24\,\mathrm{m} \times 0.24\,\mathrm{m}$ in reality. Three examples of the input representation are pictured in Figure 6.2. The goal is to predict the trajectory

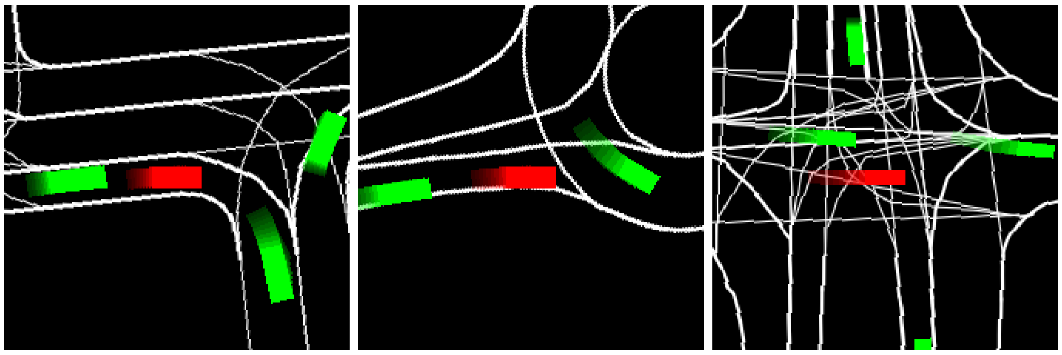


Figure 6.2: Three exemplary top-down images. The red rectangle describes the ego-vehicle whose trajectory is to be predicted. The green rectangles are other traffic participants.

of an ego-vehicle $\hat{\mathcal{T}}^i$ based on a sequence of observed history states of length $\tilde{T}$. The observed states comprise not only the information of the ego-vehicle but also that of its environment, which includes the positions and speeds of all neighboring traffic participants, supplemented by the road information. In contrast to the Relation-based Model, the additional road geometry should enable the Hybrid Model to predict a trajectory including steering angles. In Figure 6.3 the hybrid architecture is depicted. The architecture includes two parallel processing streams: the graph pipeline and the image pipeline, each of which computing an independent embedding that is combined at the end. A weighting mechanism determines the relative influence of both branches in regard to the traffic scene. The entire model is trained as a whole.

In the graph branch, the initial step involves computing an embedding for each node h^i in every graph $G(t)$ of the temporal history. This embedding contains information about the social neighborhood. The propagation of node information follows the same approach as in the Relation-based Model (Section 6.2.1, Equation (6.2)). The used message passing function has been slightly extended, so that the sum message aggregation function has been replaced by a mean operator, and the update function uses two activated fully connected layers instead of the learned weight matrix Θ as suggested in the work by Rico et al. [189]. The predicted state for each traffic participant is obtained by combining the embeddings of its historical states using a two-layer GRU network. The input size h^i is equal to the output size of ϵ_G^i after the GRU, which is a vector of size 128.

The image data $\mathcal{I}^i$ is processed by the VGG16 network [175]. A linear layer is appended to create a 128-dimensional image embedding $\epsilon_{\mathcal{I}}^i$ for the corresponding ego-vehicle i.

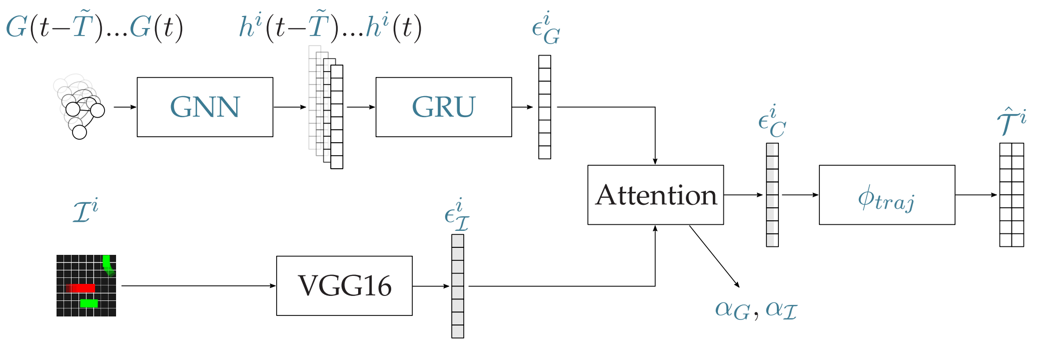


Figure 6.3: Trajectory prediction architecture for an ego-vehicle i, consisting of two parallel models for the computation of both graph ($G(t)$) and image information ($\mathcal{I}^i$). Using a learned weighting factor ($\alpha_G, \alpha_\mathcal{I}$), the resulting embeddings ($\epsilon^i_G, \epsilon^i_\mathcal{I}$) are combined. (Adapted from [5])

The attention block evaluates the importance of the two embeddings in terms of how much one of them could contribute to finding the correct trajectory. First, both embeddings are normalized to achieve an equal weighting for the subsequent task. Then, each embedding is transformed separately to a scalar by a fully connected MLP with an intermediate nonlinear activation function. The resulting scalars, α_G and $\alpha_\mathcal{I}$, are bounded to a value range between 0.1 and 0.9, and it is ensured that their sum is always equal to 1. Applying these constraints guarantees that each embedding contributes at least 10% to the final embedding. This is essential for the training process, as it ensures that a slower training of one of the two submodels is not suppressed prematurely. The combined embedding ϵ^i_C is obtained by linear combination of the weighting factors ($\alpha_G, \alpha_\mathcal{I}$) and the associated model embeddings ($\epsilon^i_G, \epsilon^i_\mathcal{I}$).

$$\epsilon^i_C = \alpha^i_G \epsilon^i_G + \alpha^i_\mathcal{I} \epsilon^i_\mathcal{I} \tag{6.3}$$

After calculating the embedding, a two-layer, fully connected MLP with a Leaky ReLU activation function ϕ_{traj} is used to generate the final position sequence $\hat{\mathcal{T}}^i$. This sequence represents the trajectory of the ego-vehicle and is composed of $\hat{T} = 30$ discrete components for a time interval of the next $3\,\text{s}$.

$$\hat{\mathcal{T}}^i = (\widehat{x^i_0}, \widehat{y^i_0}, \widehat{x^i_1}, \widehat{y^i_1}, ..., \widehat{x^i_{\hat{T}}}, \widehat{y^i_{\hat{T}}}). \tag{6.4}$$

6.2.2 Training

The training for the Relation-based Models (*Single Step*, *Recurrent*) and Hybrid Model is conducted using the same dataset. The dataset (see Section 3.1) has been divided into three sets for training purposes. The first set, which contains 10% of the data, has been reserved as a holdout set for subsequent evaluation. The second set, which represents 5% of the data, has been designated as the validation set to review the training approach in the first place and to adjust the hyperparameters for training. The largest set, which accounts for 85% of the data, has been used for the training process.

The graph representation used in Relation-based Models on its own is insufficient to accurately predict future trajectories based on input data due to the non-Cartesian nature of the processed edge features and the use of the norm of the velocity vector as a node feature. Furthermore, the neglect of road geometry further amplifies this limitation. For this reason, these two models are only trained to estimate acceleration values, as described above. The loss $\mathcal{L}_{acc}$ specified in Equation (6.5) is used for this purpose.

$$\mathcal{L}_{acc} = \frac{1}{n} \sum_{i} \left| \hat{a}^i - \tilde{a}^i \right| \tag{6.5}$$

In the Hybrid Model, a trajectory is predicted and, accordingly, the predicted trajectory $\hat{\mathcal{T}}$ is weighted with an L1 loss compared to the target trajectory $\tilde{\mathcal{T}}$ during training using the loss term $\mathcal{L}_{traj}$.

$$\mathcal{L}_{traj} = \frac{1}{n} \sum_{i} |\hat{\mathcal{T}_i} - \tilde{\mathcal{T}}^i| \tag{6.6}$$

For each entity in a scene graph, the acceleration or the trajectory is used as a training sample. It is important to note that only the acceleration or the trajectory of the traffic participants present in the scene at the starting point $t = 0$ is predicted. Traffic participants, that leave the scene earlier, are still considered for the calculation of the historical states, but not used for individual training samples.

The ADAM optimizer has been used to train all modes, with an initial learning rate of 10^{-4}. The learning rate has been periodically reduced by an order of magnitude when the validation batch fails to achieve a better loss.

6.2.3 Experiments

The models are evaluated using the Average Displacement Error (ADE_t) and Final Displacement Error (FDE_t). ADE_t represents the average Euclidean distance between the predicted vehicle positions and the ground truth over t s predicted time interval in meters. FDE_t is the displacement of the last predicted position after t s. As proposed by the INTERPRET challenge benchmark [178] the time horizon is set to $t = 3$ s. The models are compared to two naive baselines to showcase the information embedded in the SSG. The *Baseline Mean* model is a simple approach that consistently outputs the mean value across the entire test dataset. Meanwhile, the *Baseline Zero* model consistently outputs zero acceleration, also known as a constant velocity model, regardless of the input. As a result, the vehicle continues to move at a constant speed and does not react to the environment. To ensure comparability of results, a constant acceleration is assumed for future timesteps and a trajectory is computed based on the ground truth path of a traffic participant. This allows to describe the deviation of the traveled distance, which is comparable to the FDE_3 or the ADE_3 of a trajectory comparison, as follows in Equation (6.7). However, it is important to note that this model currently only predicts the longitudinal motion component (acceleration/braking) of a traffic participant and does not take into account the lateral component (steering)[*].

$$\text{FDE}_t = \left|\hat{a}^i - \tilde{a}^i\right| \frac{t^2}{2} \tag{6.7}$$

$$\text{ADE}_{\hat{T}} = \frac{1}{\|\hat{T}\|} \sum_{0 \leq t \leq \hat{T}} \left|\hat{a}^i - \tilde{a}^i\right| \frac{t^2}{2} \tag{6.8}$$

The results of the trajectory predictions are shown in Table 6.1. Compared to the baseline, the *Single Step* model is about 20% better in the task of predicting fitting trajectories for an ego-vehicle. If an ablation model without edge information, i.e., neither topology nor edge attributes, and only access the current vehicle state is considered, the *Single Step* model still performs over 10% better. This information suggests that the prediction model can estimate a trajectory (acceleration) based on a traffic scene. Additionally, the SSG stores relevant information about the traffic scene that is useful for interactive driving in traffic. This has been further investigated in the work of Grimm et al. [14]. Here, the SSG is used as additional information for a trajectory predictor and its influence on the prediction accuracy has been examined. It is been shown that the fully connected graph can be reduced to only those edges (reduction of 90%) that are also present in the SSG without

[*]Errors have been calculated using Equation (6.8) or Equation (6.7)

Model	FDE_3 in m	ADE_3 in m
Baseline Mean*	2.943	1.174
Baseline Zero*	2.696	1.075
Single Step*	2.223	0.853
Single Step no edge data*	2.484	0.953
Recurrent5*	1.219	0.434
Recurrent10*	**0.846**	**0.291**
Hybrid1	1.347	0.41
Hybrid6	**0.514**	**0.229**

Table 6.1: Trajectory prediction results on the INTERACTION dataset

any loss in performance. This suggests that the SSG contains all interactions with relevant traffic participants.

Considering the temporal network, which includes the history of the traffic scene for the prediction, the results are significantly improved. The prediction accuracy increases with the length of the history. The *Recurrent10* model, which incorporates an additional 10 timesteps equivalent to $1\,\mathrm{s}$ of history, performs better by reducing the FDE_3 by two thirds.

At first, it seems that the Hybrid Model performs worse even after including the amount of image information. However, it is important to note that the prediction task of the hybrid network is significantly more complex. This is due to the fact that the trajectory prediction requires estimation of not only the acceleration but also the steering angle. This also explains why the *Hybrid1* model performs worse than the *Recurrent10* model. The evaluation of trajectory predictors involves estimating up to 6 future trajectories simultaneously, rather than just one. Extending the model in this way allows for better comparison with the current state of the art. The ten highest-ranking models listed in the INTERACTION leaderboard [178] have FDE_3 values ranging from $0.4577\,\mathrm{m}$ and $0.6375\,\mathrm{m}$. This puts the *Hybrid6* model in the mid-range of these state-of-the-art models.

6.3 Path-following Driver Models

When imitating the explicit behavior of traffic participants, machine-learning methods are ahead of classical methods in terms of prediction accuracy [178]. However, most classical models are significantly faster than machine-learning models in terms of computation time. For an analysis of the execution time of the models, see Appendix 4. Furthermore, the classical models add new behavior patterns to the later simulation. For this reason, this thesis also examines classic, rule-based approaches in addition to machine-learning models.

One example of a simple and non-interactive behavior model is the constant velocity driver model. This model maintains the initial absolute velocity ν and follows a given path. If the initial velocity is zero, the vehicle does not move in the simulated scenario. Another non-interactive model is the emergency brake model. This driver model performs a controlled emergency stop with a given braking acceleration along a given path. The vehicle decelerates until it comes to a complete halt, where it remains for the rest of the scenario. The Intelligent Driver Model (IDM) model is a commonly used classic approach that can react to the vehicle in front. However, it does not evade obstacles in the lane, but only allows for a braking maneuver. Therefore, a predefined path must be provided for the IDM. The model aims to reach a target speed ν_0 while maintaining a distance d_F^*. The acceleration a is determined by an expression that incorporates the current velocity ν, and the gap to the vehicle in front d_F. The IDM is highly parameterizable, whereby the maximum acceleration a_{max} of the ego-vehicle, the comfortable braking deceleration a_b, as well as the minimum distance to the front vehicle d_{F0} and the minimum time to the front vehicle t_0 can be set in order to achieve the desired driving behavior. The acceleration coefficient δ is typically set to 4, but it can also be used to weigh the importance of the corresponding velocity based on the required use case.

$$a = a_{max} \left(1 - \left(\frac{\nu}{\nu_0} \right)^{\delta} - \left(\frac{d_F^*}{d_F} \right)^{2} \right) \tag{6.9}$$

$$d_F^* = d_{F0} + \nu \cdot t_0 + \frac{\nu \cdot \Delta\nu}{2 a_{max} a_b} \tag{6.10}$$

The relation-based and rule-based models described above are not capable of independently determining lateral acceleration, and are therefore dependent on an externally specified path. This work examines two possible methods for determining the path. Firstly, the trajectory data can be extracted from the dataset. However, this method requires the traffic participant traveling exactly on the recorded path in order to continue using it. However, one challenge is that the predefined path is finite and ends as soon as the recording is finished. This implies that vehicles included in the data set cannot be assigned to a trajectory in the simulation. An alternative option is to follow the center line of the current lane. The shape of the centerline of a lane defines

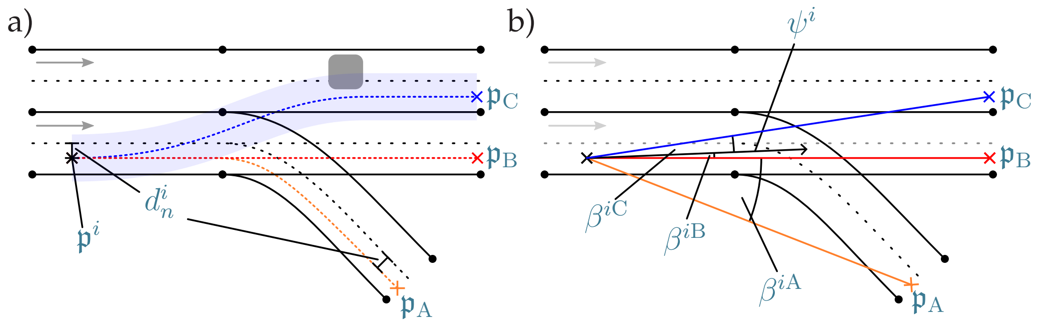

a)
$\mathfrak{p}_\mathrm{C}$
$\mathfrak{p}_\mathrm{B}$
d_n^i
$\mathfrak{p}^i$
$\mathfrak{p}_\mathrm{A}$
b)
ψ^i
$\mathfrak{p}_\mathrm{C}$
$\mathfrak{p}_\mathrm{B}$
$\beta^{i\mathrm{C}}$
$\beta^{i\mathrm{B}}$
$\beta^{i\mathrm{A}}$
$\mathfrak{p}_\mathrm{A}$

Figure 6.4: Schematic illustration of possible paths for a traffic participant along the lanes (a). Heuristic for determining the most plausible paths in regard to the orientation ψ (b)

the course of the path. The starting point for determining the respective path is always the center of the traffic participant. In most cases, the paths run parallel to the respective centerlines of the lanes at a constant spacing. An exception to this is a lane change to parallel lanes (see blue path in Figure 6.4). It is possible for a traffic participant to have several possible lanes to follow. To select the lane, an anchor point is calculated for each option, which is defined by a length when following the respective lane. Figure 6.4 displays three potential lanes for which the anchor points ($\mathfrak{p}_\mathrm{A}, \mathfrak{p}_\mathrm{B}, \mathfrak{p}_\mathrm{C}$) are calculated. The next step involves selecting the path with the smallest angular deviation ($\beta^{i\mathrm{A}}, \beta^{i\mathrm{B}}, \beta^{i\mathrm{C}}$) between the orientation of the traffic participant ψ^i and the vector between the center point and the anchor point. In principle, this step can be parameterized in several ways, e.g. it is always possible to try to turn in one direction, whereby the path with the largest angle in one direction is selected. Furthermore, a collision calculation is performed using this path, in order to determine, for example, the distance to the next relevant traffic participant for the IDM. The distance between each traffic participant and the path is calculated. When the distance falls below a predefined threshold value (see gray box with the blue path in Figure 6.4a), the traffic participant is considered an obstacle to which the model must react to. A clearance of 5 m is uniformly set for all traffic participants in this work.

6.4 Summary

This chapter introduces and implements two types of behavior models that are used in Chapter 7 to simulate realistic and reactive behavior in a simulation environment. The focus here is on imitation learning-based trajectory predictors. These have the ability to process the entire scene and all the situations implicitly contained in it, and to predict a trajectory that is as close as possible to real human behavior. The trajectory of the developed Hybrid Model mimics the behavior of the respective humanoid driver on average to 0.41 m. In combination with rule-based behavior models, which are already widely employed in simulations, the presented behavior models and prediction models address the research questions outlined in Research goal 5, thereby demonstrating how a behavior is modeled to represent realistic driving behavior.

7 Resimulation

Simulation has emerged as a powerful tool for conducting scenario-based testing in the area of traffic scenarios. The accuracy and validity of a simulation depends on the expression of real behavior of the entities participating in the traffic scenarios. It is very difficult to have a hybrid simulation where recorded trajectories of traffic participants and intelligent actors coexist, especially in more complex traffic scenarios [60]. Consequently, it is essential to employ a dynamically responsive testing environment to assess and compare the driving capabilities of various HADFs. Consider a simulation of a scenario involving a merging actor vehicle under event and time-based control. Here, the actor vehicle starts moving as soon as a certain point is reached by the System under Test (SuT). For conservative driving HADFs with a low target speed, the situation may be deemed critical in one instance, while another HADF with a high target speed, having already passed the on-ramp. Due to the different behavior of HADFs, it is possible that a desired situation of a test scenario is hardly or not at all evoked. This renders test scenarios ineffective. This is especially relevant when other traffic participants must actively react to the ego-vehicle.

One challenge is that direct access to a HADF is not always available (see Chapter 1), and there is a need to identify **generally** challenging traffic scenarios.

This chapter introduces an alternative approach to conventional methods, aiming to uphold independence from the tested driving function. Instead of relying on predefined scenarios, seed-scenes, which can be any frame of a data recording, are employed to exclusively define the initial conditions of each scenario. The subsequent evolution of the scenario can be determined by the driving function to be tested and the decisions that result from it.

To gain insight into the criticality of a seed-scene, potential outcomes are extrapolated using closed-loop simulations using behavior models of the actors involved. This process allows for a comprehensive evaluation of different simulated futures, ultimately providing a holistic assessment of a scene's significance in testing HAVs.

Parts of this chapter have been published previously in the following peer-reviewed publications [3], and some of the content has been adopted verbatim.

7.1 Related Work

7.1.1 Scenario Generation

As stated in Section 2.1.4, scenario-based testing is a main approach for testing HAVs. The literature presents various approaches for generating test scenes or scenarios to validate the SuT in a test process.

Section 3.2.2 presents ontologies for describing scenarios. Scenarios can be generated by combining different classes and sampling from a parameter distribution [74, 75, 190]. After selecting a logical scenario, scenario exploration can be used to generate particularly critical scenarios, for example, by searching through scenario parameters [45]. Furthermore, languages have been developed specifically for describing traffic scenes and scenarios. One popular description format for simulation is OpenScenario [108]. Scenic [191] is a probabilistic programming language that allows manual creation of traffic scenes. Object positions and attributes can be defined as distributions, which are then automatically sampled to generate a variety of scenes. The work of Klischat et al. [192] addresses the problem of needing traffic recordings by generating scenarios using a language. They define scene specifications that result in a scenario. Trajectories and lane changes are determined by solving an optimization problem. In the work of Li et al. [193], initial traffic scenes of a simulation are generated by randomly spawning traffic participants at given spawn points, which are then controlled by rule-based behavior models, such as an IDM. However, a focus of the work is the generation of novel scenarios in which different road courses are always generated by procedural generation.

Manual and rule-based methods may not accurately represent the complexity and diversity of the real world. To address this issue, Tan et al. propose an autoregressive approach in their work SceneGen [52]. This method iteratively adds various traffic participants to the scene, starting from an initial traffic scene. The work of Feng et al. [194], titled TrafficGen, builds upon the principles of SceneGen. The authors develop an autoregressive machine learning model that enables scene augmentation or even new scene generation. Additionally, a trajectory prediction module is used to further predict an initially generated scene to obtain a scenario. The generated scenes are integrated into a reinforcement simulator to train a behavior predictor. The other traffic participants in the generated scene are controlled using IDM and lane change models. The methodology used in TrafficSim [195] is similar to the one described in this chapter. TrafficSim aims to generate realistic multi-agent behavior automatically. To achieve this, a combined latent scenario representation is created for all vehicles based on a map and scene representation. This is then used in combination with the states of the individual traffic participants to sample different futures. A significant difference in their work is that the joint behavior of traffic participants is described by a single learned model. Therefore, the behavior of all traffic participants is inherently known by the model.

It is worth noting that, with a few exceptions [194], previous works have primarily focused on developing generative models. Subsequently, a scenario is generated using (rule-based) models. This thesis will further investigate the latter step of the methodology.

7.1.2 Simulators

Simulators have been extensively studied as valuable tools for the development and training of HADFs. Various simulators are available for different applications, taking into account specific features and environments. These range from macroscopic traffic flow simulations, such as the SUMO simulator [169], to physically based real-world simulations (CARLA [196]). In recent years, numerous simulators, particularly in the field of computer vision, have been developed. Notable examples of simulators with a high degree of graphic realistic traffic scenarios include GTA V [197], known for its immersive open-world environment; CARLA [196], known for its realistic urban environments and convenient programming interface; and its extension SUMMIT [198], designed to further enhance simulation capabilities. These simulators are adept at generating 3D traffic scenarios, making them particularly suitable for comprehensive testing of highly automated systems, encompassing sensor perception. The realistic environments created by these simulators enable researchers and developers to evaluate the performance of HADFs under various conditions, contributing to the refinement and optimization of autonomous driving technologies.

In the context of highly automated driving, the latest research in simulation places significant emphasis on object lists. Prominent examples are BARK [60], Nocturne [199], nuPlan [50] and Waymax [200], all of which simulate traffic scenarios in a closed-loop manner. These state-of-the-art simulators are able to support multi-agent simulations by using recorded input data and employing a 2D coordinate system to ensure a lean and lightweight design. The primary objective of these simulators is to represent the dynamic parameters present within a traffic scene. They demonstrate a high degree of compatibility with the input data format utilised in this work (see Section 3.1), as it is precisely these parameters, such as the development of the position of the traffic participants over time, that are examined. Furthermore, the road information is used to animate the traffic scene with the help of various simulation models.

It should be noted that the main focus of the environmental modeling developed in this thesis is on evaluating the behavioral aspects of both the ego-vehicle and other vehicles within the simulated scenarios. In contrast to prioritizing the perceptual capabilities of HAVs, this approach does not require sensor modeling. The simulators under consideration focus on providing behavior-specific features, such as the exact position, size, and path of vehicles moving within the simulated road network. This emphasis on behavioral complexity contributes to a more sophisticated and targeted analysis of vehicle interactions within this simulation framework.

7.2 Simulator Concept and Implementation

In this work, a sophisticated multi-agent simulator will be used to extrapolate scenarios based on real seed-scenes. Recent advances in simulation technologies and benchmarks have significantly influenced the development of the simulation environment implemented in this thesis. Various implementation and concept paradigms of the 2D simulators mentioned in the related work chapter (e.g. Nocturne [199], nuPlan [50] and Waymax [200]) are applied and further developed in the simulation environment presented in this work.

The simulator functions as an execution unit, passing an input scene to various behavior models and updating calculated trajectories to the next state after each step. The interfaces between modules are standardized, allowing for module exchange as desired. Other simulators, such as those presented in Section 7.1.2, can also be employed if additional aspects of the environment need to be simulated. Assuming a physics calculation is required, CARLA [196] may be used instead of the simulation framework implemented for this work. As stated previously, this work focuses solely on the actions of traffic participants and the resulting trajectories, which, depending on the model, can be calculated using object lists and road maps exclusively. Therefore, the simulation can be implemented in a lightweight manner, which accelerates the simulation time (see Figure A.10).

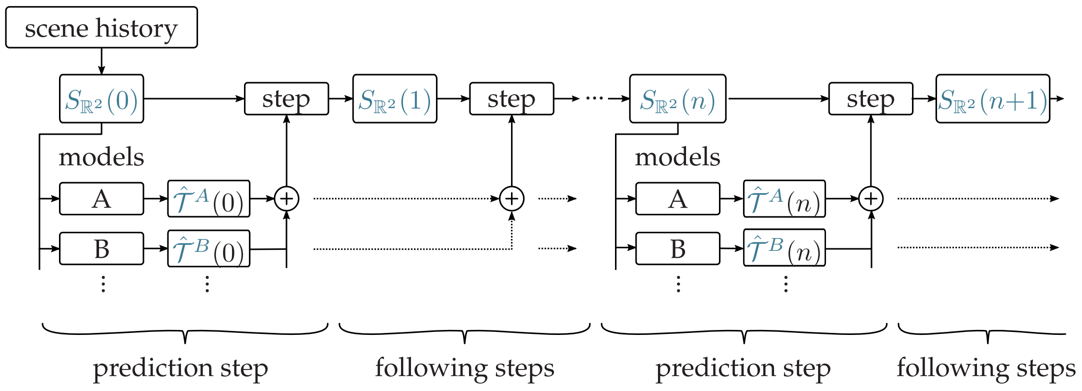


Figure 7.1: Block diagram illustrating the simulation process. Trajectory proposals are computed from a seed-scene ($S_{\mathbb{R}^2}(0)$), considering various behavior models based on the current situation. Each trajectory is followed for a certain timespan (following steps) before a recalculation is performed, taking into account the changed scene context (prediction step). (Adopted from [3])

Figure 7.1 illustrates the sequence of simulation steps. The user specifies a traffic scene to be extrapolated, the so called seed-scene. The choice of the seed-scene can of course be selected by certain heuristics or randomly sampled from a dataset. The initialization process also involves assigning different behavioral models to the individual traffic participants in the seed-scene. These models can be assigned manually for a particular outcome, or randomly using a predefined distribution function.

The simulator provides environment and context information for individual models. In line with current research in trajectory prediction, it also includes history scenes from the previous second. The history facilitates the usage of additional imitation learning models (see Chapter 6). The simulator shares the same environment for each trajectory computation step, allowing the models to be computed in parallel. This environment includes the position, size, velocity, and classification (type) of all traffic participants for the current and previous 10 time steps*. Furthermore, a high-definition road map that contains the geometry and topology of the road is also provided. The state representation is built upon the scene representation of the dataset (see Section 3.1) and its specifications. This applies not only to the spatial component but also to the temporal component. Therefore, the simulated duration of a simulation time step is 100 ms (10 Hz). This ensures sustainable reusability and the use of other tool-kits [85, 86, 6]. The choice is made to look exactly

*Common in trajectory prediction and the history length in the INTERACTION dataset utilized in this work like the INTERACTION dataset. This allows consistency from model training to inference, i.e. application as a behavior controller.

The models calculate a full trajectory (3 s) for each of the assigned traffic participants. To enhance simulation performance and the trajectories' continuity, trajectory calculation can be skipped for a configurable number of steps n, enabling actors to follow pre-calculated trajectories for a brief period. In the simulation sequence in Figure 7.1, this is labeled as *following steps*. The simulation produces best results when trajectories are calculated every five to ten steps. Compared to a higher prediction frequency, the predicted trajectories between consecutive timesteps are more stable and the simulation speed increases. If too much time elapses between predictions, the models may not be responsive enough to react to obstacles, for example. These findings align with those presented by Gulino et al. [200]. After each step, the states of all actors are consolidated and recorded in the data logger. The data logger virtually records the history and makes it available for later analysis.

It is important to note that the number of traffic participants in a scene remains constant throughout the entire scenario. Therefore, no new traffic participants are added after the seed-scene, unlike some real-life recordings. Additionally, while vehicles can leave a region of interest, they continue to be simulated in the simulator.

7.3 Simulation Experiments

In order to gain a general understanding from multiple perspectives and to examine a variety of outcomes, a number of child-scenarios are simulated. Afterward, the experiments aim to analyze the criticalities of the traffic scenes described by the road geometry and encountered traffic participants. Figure 7.2 illustrates the relationship between the seed-scene and a child-scenario, and also visualizes the evaluation of the experiments in a schematic drawing.

The six different behavior models defined in Chapter 6 are used to control individual traffic participants. Table 7.1 provides a list of specific behavior models that can be selected for simulation runs. The selected models are intended to depict the behavioral spectrum of traffic scenarios and actively trigger reactions from traffic participants. The first five models are based on the path-following model, where only the acceleration behavior is defined by the model itself. The path to be followed is determined by the initial position on a lane and the resulting route (shown in Figure 6.4). The *Standard Driver* and *Risky Driver* models are based on the IDM (refer to Equation (6.9)). The

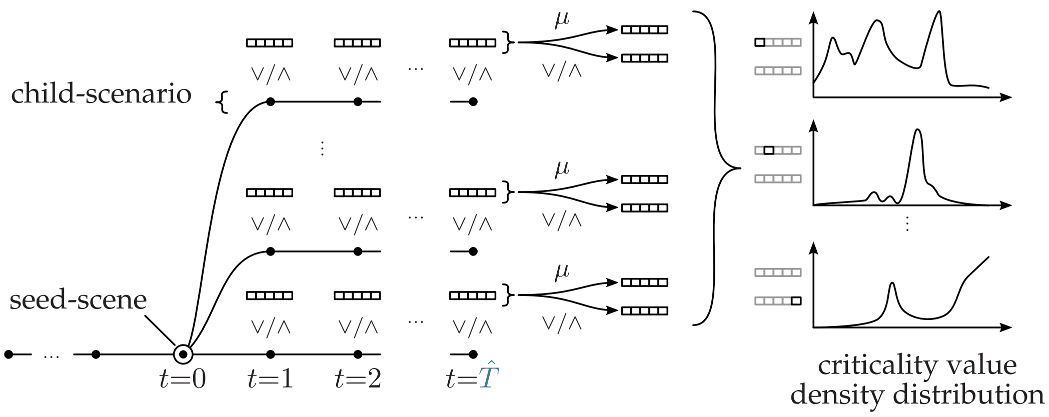


Figure 7.2: Schematic illustration of the simulation experiment process. A seed-scene ($\odot$) yields to several future scenarios (child-scenarios). Each child-scenario consists of $\hat{T}$ consecutive traffic scenes ($\bullet$), which are evaluated using a criticality analysis to determine the criticality extrema $\vee/\wedge$. The criticality values of each scene of a child-scenario are then aggregated into two scores - one by finding extrema ($\vee/\wedge$) and the other by calculating the mean values (μ).

standard driver represents a moderate driver. The parameters are inspired by Treiber et al. [167] but have been slightly modified to better reflect the general behavior of the trajectories in the dataset. The risky driver accepts smaller distances to the vehicle in front and is expected to have a higher acceptable acceleration. The *Constant Velocity* model maintains the initial speed specified in the seed-scene. This driving behavior can be compared to a driver who is briefly distracted and therefore unable to react to their surroundings. The *Emergency Brake* model performs an abrupt braking maneuver with a constant acceleration of $-5.0\,\frac{\text{m}}{\text{s}^2}$. The *Relation-based* model represents a learned average behavior of the dataset, depending on the given traffic situation. It is an interactive model that includes the relevant dynamic objects in the traffic scene in the calculation of the behavior. The *Hybrid1* model increases the degrees of freedom by extending it with the steering angle, i.e., the choice of the followed path. Machine-learning behavior models use nine additional time steps from the past for initialization in addition to the information in the seed-scene.

The duration of a child-scenario simulation is 3 s, which equals to $\hat{T} = 30$ time steps. After each simulation run, the seed-scene is reinitialized by randomly sampled behavior models to generate the next child-scenario. The probability of assigning the models is uniformly distributed.

Table 7.1: Models used in the Simulation

Name	Parametrization
Standard Driver (6.3)	$a_b = 1.7$, $T_0 = 2.8$, $d_{F0} = 5.0$, $\delta = 4$
Risky Driver (6.3)	$a_b = 3.0$, $T_0 = 1.7$, $d_{F0} = 2.0$, $\delta = 4$
Constant Velocity (6.3)	Follows path with initial speed
Emergency Brake (6.3)	Decelerates abruptly at $-5.0\,\frac{\text{m}}{\text{s}^2}$
Relation-based Model (Recurrent10) (6.2)	Acceleration imitation model
Hybrid1 Model (6.2)	Trajectory imitation model

After the simulation process, each simulated scene in every child-scenario is evaluated objectively based on criticality using the metrics described in Section 4.2. Since the criticality values are only compared later in the process by their distribution but not directly compared in a figure (see Figure 4.2), the sensitivity adjustment from Equation (4.4) is not applied. The most critical value for each scene and metric is stored, resulting in a score for each traffic scene based on the smallest values for Dist, GT, PTTC, and WTTC, and the largest values for inverse TTC (TTC*) and TQ. In summary, the nanoscopic metrics are aggregated to determine microscopic scene criticality [133]. To evaluate a complete scenario, the extrema for all scenes of a child-scenario are picked. Additionally, to mitigate the impact of outliers, mean values of all microscopic scene criticalities are summarized. In conclusion, a criticality value is calculated for each pair of vehicles (for binary metrics) in a scene, based on the six aforementioned metrics. Subsequently, the extrema are filtered in order to obtain a criticality value for each metric in each scene, resulting in six values. This process is repeated for each scene in the child-scenario, resulting in $6 \times \hat{T}$ values. On the one hand, the extrema of the $6 \times \hat{T}$ values are extracted ($\equiv 6$ values), and an average value is calculated for each metric ($\equiv 6$ mean values). As a result, each child-scenario is described based on its criticality with 12 corresponding values (compare Figure 7.2).

The next step is to consolidate these criticality values for each simulation run and all corresponding child-scenarios. As the values change depending on the constellation of the simulated behavior models, the resulting criticality values, considered individually for each criticality metric, are distributed over a wide spectrum (distribution of criticality values). A Kernel Density Estimation (KDE) is utilized to display the distribution of criticality values for all child-scenarios. The KDE enables comparison of continuous value spaces and smooths the density distribution, providing comparable results when a number of simulation runs are sampled. A Gaussian kernel with a bandwidth of 0.1 is used for smoothing on all metrics.

7.3.1 Sampling Size

In order to investigate the extent to which the extrapolated future (child-scenarios) can provide a statement about that of a seed-scene, the analysis begins with an examination of the number of iterations for the subsequent simulation runs. The number of possible futures increases exponentially as the number of traffic participants present in the seed-scene increases. This is denoted by the various assignments of all used models to each vehicle in the scene. Given an average of 11 traffic participants per scene in the used dataset and considering 6 different models, the simulation requires on average of 3.6×10^8 iterations to cover all possible futures within this configuration. Determining the necessary number of simulation runs to approximate all possible cases is crucial due to the impracticality of computing such a magnitude within reasonable time constraints, even only for a few scenes.

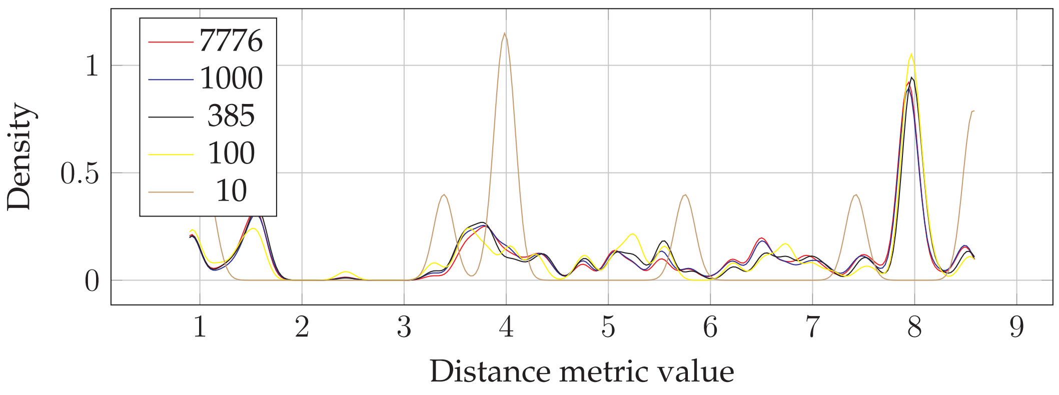


Figure 7.3: Comparison of density functions for different sample sizes in regard to the characteristic coverage of the distance metric. (Adapted from [3])

The density functions of the distance metric in a scenario with five traffic participants (small population) are depicted in Figure 7.3. Simulations are conducted for every possible combination of models and traffic participants, resulting in a total of 7776 runs[†]. Afterward, 1000, 100, and 10 child-scenarios are randomly selected. A set of $n = 385$ runs is also chosen based on the sample size estimation approach [201]. The sample size n is calculated with a standard confidence level of 95%, resulting in a standard score of $z \approx 1.96$. A margin of error ε of 5% and the population size as unknown is assumed.

[†]As this is a traffic scene with very few traffic participants, a complete analysis of the possible child-scenarios is possible.

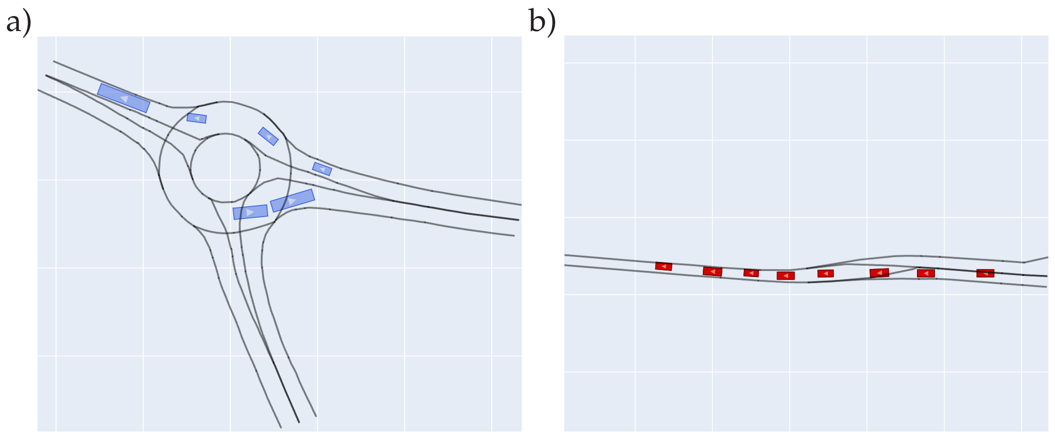


Figure 7.4: Example seed-scenes used for the evaluation in Figure 7.5. A scene on a roundabout (a) and a scene on a merging road (b).

The sample size is calculated using Equation (7.1) (assuming a population proportion of 0.5):

$$n = \frac{z^2 \cdot 0.25}{\varepsilon^2} \tag{7.1}$$

Upon examining the correspondence of the curves, it is clear that a smaller sample size leads to a greater deviation in the shape of the density distribution, which is correlated with the representativeness of the sample size. To optimize computational efficiency, the objective is to choose the smallest practical sample size. Therefore, all subsequent simulations are conducted with 385 runs. This enabled the representation of important features and significantly decreases the computational time.

7.3.2 Scene Criticality Analysis

In this section, the results of the simulation are discussed in the context of criticality and its correlation to the seed-scene. Figure 7.5 shows the smoothed density distributions of the analyzed metrics for two seed-scenes in blue and red (all other plots can be found in the Appendix 2). The blue scenario is taken from a roundabout map (Figure 7.4a), while the red scenario is taken from a highway entrance (Figure 7.4b) with a merging situation. The ground truth[‡], which displays the actual course of the scenario as recorded in the

[‡]As for each seed scene, the real sequence as it is actually contained in the recordings (dataset) is referred to as ground truth

dataset, is marked by a dashed line. It is noteworthy, that the density distributions of the criticality distributions rarely correspond to a normal distribution. Instead, they exhibit distinct characteristics depending on the seed-scene. It is apparent that the criticality values of the ground truth do not always accurately reflect those of the seed-scene. For example, criticality metrics cannot be calculated for the ground truth scenario depicted here, but they can be computed for some child-scenarios. Some metric results have less than 385 calculated criticality values because they cannot be computed in certain child-scenarios. GT is not applicable in the merging scenario (red) because it is a metric for intersection trajectories. In this scenario, only car-following constellations exist. As shown in Table 4.1, the applicability of various criticality, such as GT, is not given in every scene. With the criticality distributions shown here, but also with most other distributions, it becomes evident that the criticality value of the ground truth (recording from the dataset) does not provide any direct information about the seed scene and how it develops. Accordingly, the ground truth markings are arbitrarily positioned in the distribution. This demonstrates the value of resimulation in showing that small deviations in the behavior of traffic participants can lead to significantly different scenario outcomes, especially in terms of criticality.

The orange line indicates the suggested criticality threshold for shown metrics in the literature. Child-scenarios that exceed the criticality threshold are highlighted. In case of Dist and GT, all child-scenarios on the left side of the line are critical. For the TQ and the TTC*, all simulations on the right are critical. The following thresholds are used: Dist 5 m, GT 0.5 s [142], TQ 1.2 (derived from evaluations in [10]), TTC 1.5 s (TTC* $\equiv 0.667\frac{1}{s}$) [136]. At this point the Criticality Potential (CP) is introduced. The CP of a scene can therefore be defined using the integral of the density distribution (see Definition 4), where either only the definition range below or above the threshold is taken into account, depending on the metric (see highlighted areas in Figure 7.5).

With respect to Dist metric, the red seed-scene leads to more critical child-scenarios than the blue one. However, this can be explained by typical shortcomings in the Dist metric. Drivers often travel side by side in different lanes during merging. This results in small distances and therefore high criticality. Typically, human drivers do not consider this to be a critical situation. The TTC* (mean) metric for the red curve also shows a more critical scene than the blue curve. Again, this is because TTC* can only be calculated for car-following constellations, and the red seed-scene mainly consists of car-following combinations. By examining the TQ, it is noticeable that the red seed-scene has more child-scenarios above the threshold. On the other hand, the blue scene has some critical scenes with very high TQ-values.

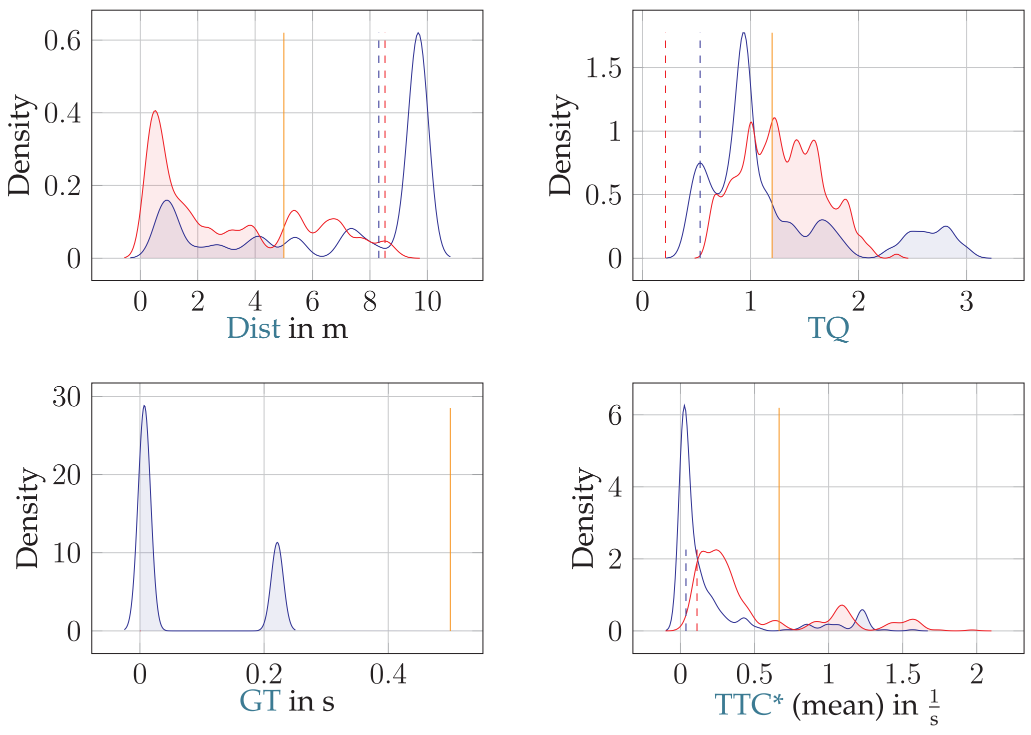


Figure 7.5: The blue (Figure 7.4a) and red (Figure 7.4b) plots show the distribution of criticality for all child-scenarios. The dashed lines mark the measured worst cases in the real situation respectively. If there is no dashed line, the metric can not be calculated for this seed-scene. GT can not be calculated for the red scenario.

Definition 4. The Criticality Potential (CP) of a seed-scene is the percentage of simulated child-scenarios which exceed a given criticality criterion.

Nevertheless, the criticality values are not further weighted in terms of CP, but only in terms of their position relative to the threshold.

Appendix 5 discusses the influence of a behavior model trained exclusively on critical scenarios on the evaluation of the seed scene, thus providing an indication of the general informative value of the criticality distribution.

7.4 Test-Space Analysis

In this section, the result of the clustering procedure is combined with that of the resimulation. The clusters and the test space are examined with regard to the criticality potentials of the individual traffic scenarios. Since the embedding space is 12-dimensional, the Uniform Manifold Approximation and Projection (UMAP) method is used to reduce the high-dimensional space to a 2D space. It should be noted that the spatial mapping of UMAP is not deterministic. In particular, the global topology of the embedding space cannot necessarily be mapped accurately. Although the UMAP method has advantages over the t-distributed Stochastic Neighbor Embedding (t-SNE) method in this respect, it is not perfect. Therefore, although local clusters can be assigned a deeper meaning, the global form of the mapped space is not necessarily significant. However, since the focus of the subsequent analysis is primarily on the local clusters, the disadvantages of the UMAP are negligible[§].

Figure 7.6 shows four scatter plots of the test space. Each point represents a traffic scene selected at random from a total of 3636 scenes (holdout set from clustering in Chapter 5), each of which is sampled with equal probability, and the color indicates the CP of the scene with respect to one of the four metrics shown as examples (Dist, TQ, GT, TTC*). Larger illustrations of the clusters can be found in Appendix 1.

In general, it can be observed, that the coverage in the test space, or the specificity, is high. Traffic scenes with a high CP are found almost uniformly throughout the entire space.

The data points can now be grouped according to their position in the UMAP-space using a hierarchical clustering method (see Chapter 5). To evaluate each cluster, the criticality potentials of all 12 criticality metrics are calculated for all child-scenarios. The average of all CPs of the metrics of all scenes within a cluster is then determined. These averages are presented in Table A.1 for the 32 clusters found within the scene space. In addition, the overall value of a cluster is shown, which summarizes the average of all CPs of a cluster.

Looking at the universal metrics (Dist, TQ, WTTC) in Table A.1, one notices a common trend that the CPs of these metrics correlate with each other. Furthermore, as expected, the CPs of the intersection-focused metrics (GT and PET) are highly correlated. Cluster 15 (cluster centroid at approx. [10.9, 6.3] in Figure 7.6) is interesting, where all CPs except PTTC are maximum, where on average 41.9% of the child-scenarios are critical. Figure 7.7a shows example scenes of this cluster. In particular, this cluster contains scenes with many traffic participants. As a result, the traffic density is generally higher on average than in other scenes with fewer traffic participants. This increases the Dist value and the TQ. At the same time, the probability that one of the traffic participants exceeds a threshold is increased. This is due to the

[§]The 2D-shape of the local clusters may differ, but the data points they contain rarely fluctuate fact that, when determining macroscopic criticality, the maximum value of a scene, and ultimately the scenario, is always selected. However, this cluster also contains more traffic situations that can potentially lead to a collision. In the first scene, a vehicle is turning left, but the distance to the crossing vehicle, which is traveling at a relatively high speed, is small. A similar situation can be observed in the second scene. The vehicles have a relatively high speed, but the lateral and longitudinal distances to the neighboring vehicles are very small when merging. The third scene shows a shortcoming in the simulation. This scene is considered very critical. This is due to the fact that many accidents occur between the two upper vehicles, i.e. the vehicle on the upper entrance to the roundabout and the parked vehicle. Parked vehicles, especially those parked outside the road, are comparatively rare in the dataset. The simulation does not consider the vehicle outside the lane as parking and which should not be moved. However, motion models are applied so that the vehicle attempts to get back into the lane. A similar phenomenon occurs with the vehicles at the bottom right, but the parked vehicle is still within the lane boundary and can be better controlled by the motion models. Here it becomes clear that the underlying simulation focuses on traffic participants road bound and can simulate traffic participants that are off the road with low reconstruction quality.

Three scenes from Cluster 6 (cluster centroid at approx. [18.6, -6.6] in Figure 7.6) are shown in Figure 7.7b. As with the aforementioned cluster, this cluster also exhibits markedly high CP values, yet it is populated by a significantly smaller number of traffic participants. The roundabout (OF) is particularly well represented in this cluster. The critical situations that occur at this location are primarily associated with vehicles that are just entering the roundabout. In many cases, the right of way is taken or the distance to the intersecting vehicle is small.

The traffic scenes shown in Figure 7.7c are taken from Cluster 4. This cluster (cluster centroid at approx. [-0.2, 2.7] in Figure 7.6) has the lowest values for all metrics and the overall CP. When examining the scenes, it is noticeable that scenes with low traffic density and few traffic participants prevail. Furthermore, most of the traffic participants have a low initial velocity. In the clusters shown in Figure 7.7, the connection between the CP and the initial traffic scene is apparent, and semantic similarities between the scenes within each cluster can be recognized. This becomes more difficult when a random cluster is in the mid-range of criticality. In this case, it is difficult for a human to find clear similarities. As stated by Hauer et al. [51] this is a foreseeable behavior of automated clustering.

This inconsistency of individual clusters can also be seen in the variance of the CPs within such a cluster. There are individual data points (seed-scenes) that have a particularly high CP, while scenes in the immediate neighborhood are barely remarkable in terms of CP.

In summary, clusters that are at the ends of the CP spectrum, with very high or very low CP values, also have this property for most of the seed-scenes in the corresponding clusters. However, there are also clusters where this is not the case, and the average CP of a cluster is not representative of many scenes within that cluster.

7.5 Summary

This chapter describes the resimulation of traffic scenes and illustrates how this can be utilized to make a general assertion about the initialization of a traffic scene in the context of testing HADFs. Through the execution of numerous simulation runs with varying permutations of driving behavior, a profile of a traffic scene is created. This profile can be employed to identify potentially critical traffic scenes. The combination of this information about the CPwith the clusters presented in Chapter 5 allows for the identification of groups of scenarios that may be particularly relevant for testing an aspect of a HADF.

The results presented here conclude the methodology of creating a test pipeline by integrating the previously presented results and artifacts, including clustering, criticality analysis, and the application of behavior models. This integration provides a final statement on the initialization of a processed scene in the context of relevance for testing.

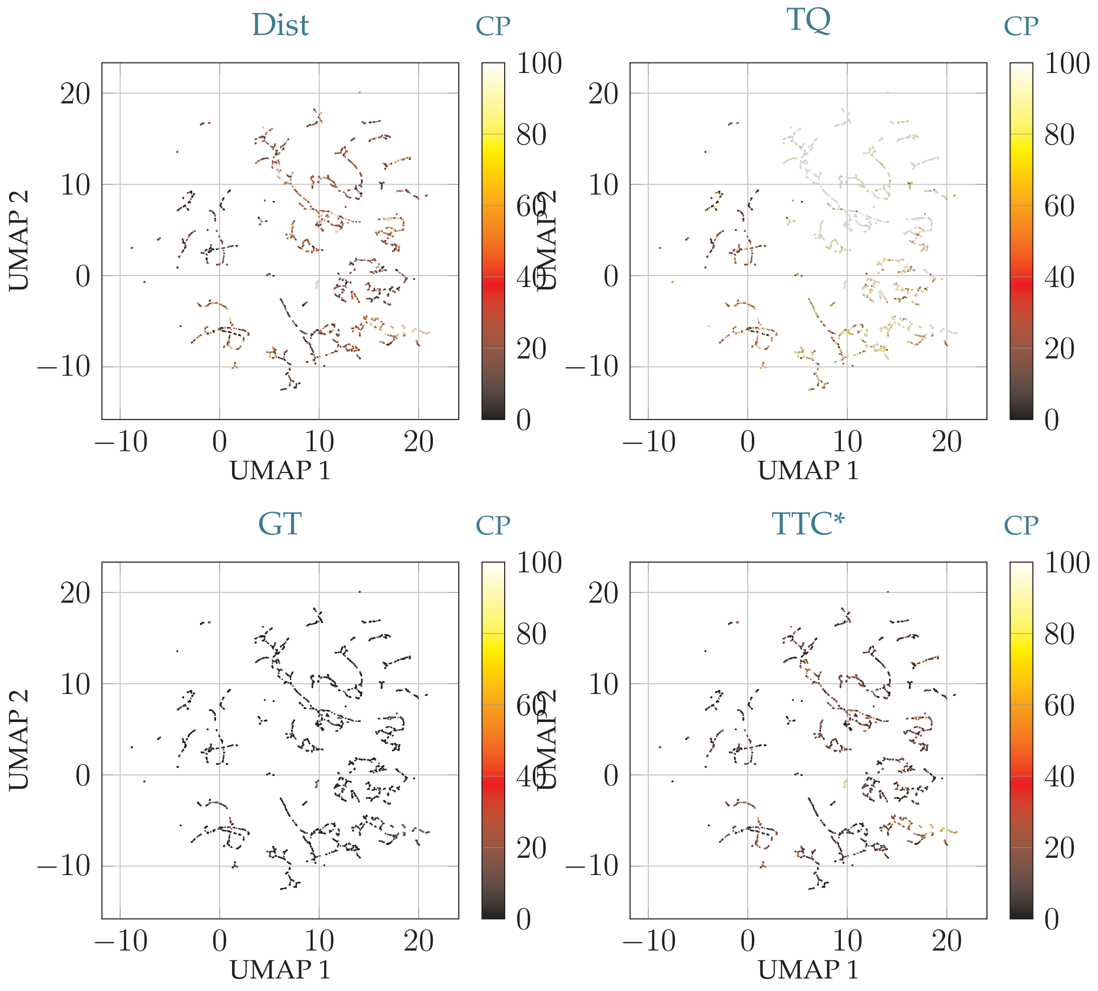


Figure 7.6: Four embedding spaces, each colored according to the respective criticality metrics (Dist, TQ, GT, TTC*), have been reduced to two dimensions. A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP.

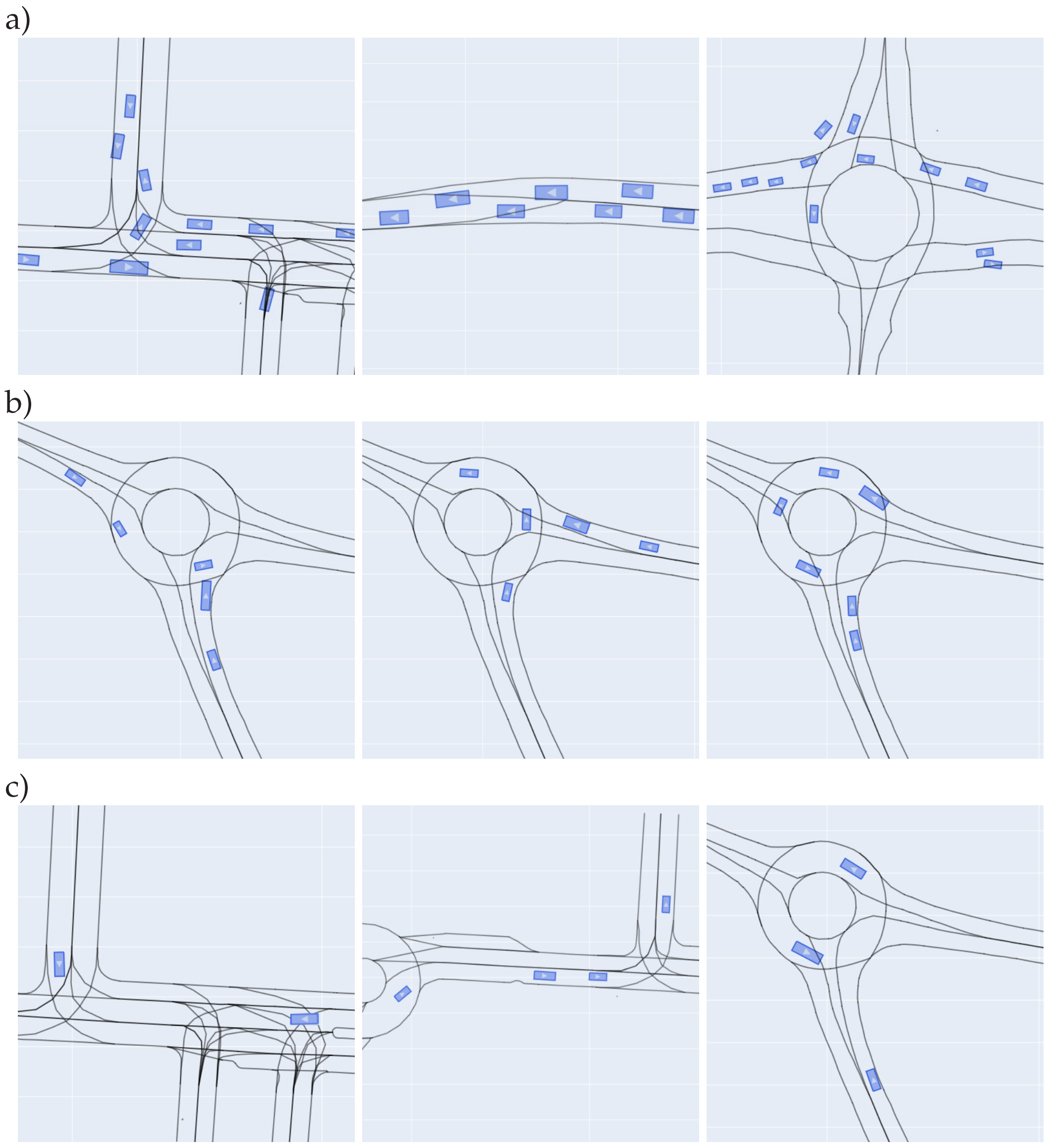


Figure 7.7: Illustration of three exemplary traffic scenes for a cluster with low CP (c), with high CP (a, b)

8 Conclusion and Outlook

This chapter summarizes this work and highlights key contributions. The contents are put into context of the research questions. Furthermore, the applicability of the results is analyzed, and the resulting suggestions for further research are discussed.

8.1 Summary and Contribution

For the widespread introduction of highly automated driving in public transport, the verification and validation of such systems is a crucial prerequisite. Road traffic is an interactive environment in which the actions of traffic participants have a significant impact on other traffic participants and their decisions. This can be observed particularly in the trajectories of individual traffic participants. Accordingly, closed-loop scenarios are of great importance for testing, in which all traffic participants react to an unknown System under Test and adapt their own trajectories accordingly. In addition, the scenarios and framework must satisfy a number of further criteria. Firstly, the scenarios must exhibit high **fidelity** and closely mirror real-world situations. Secondly, only the most relevant scenarios are to be tested in order to ensure the most **efficient** use of testing resources. Moreover, not only is a single system test to be conducted efficiently, but the entire framework needs be applicable to a wide variety of SuTs in order to ensure **transferability** and to avoid the development of new test frameworks for each system. This leads to the central research question (Research goal 1):

> *What does a pipeline look like that generates test scenarios for the Highly Automated Driving Function under test as a black box ?*

This represents the central research question, which is addressed by the individual submodules throughout the entirety of the work laid out in this thesis. To this end, this thesis presents a framework that facilitates the identification of important scenarios for testing Highly Automated Driving Function. The initial condition, more specifically, an initial scene that can be used for a subsequent test, is grouped and evaluated for its criticality. The individual aspects of this overarching research objective are divided into four more specific parts, with each part of the thesis contributing to the overall result by answering smaller sub-research questions.

The first part of this thesis addresses the scene description, which serves as the interface for incoming data for this framework. It considers the following research question (Research goal 2):

> *How can traffic scenes be described independently of the underlying road geometry?*

A traffic scene is an abstract concept and can represent various aspects of traffic. In the course of this first part, relevant attributes and important properties of a traffic scene are identified, which play a particularly important role in the context of dynamics and future behavior. To answer this question, the Semantic Scene Graph (SSG) is introduced, which represents a central contribution. The SSG reduces a traffic scene to the traffic participants it contains. In the course of this, a graph model is derived that maps the relations between the traffic participants based on their relative position within the road topology. With additional attributes that specify the relations, this description model is able to define scene constellations and make traffic scenes comparable regardless of the respective road geometry.

The second part of the work builds on the scene description of the first part and automatically assigns similar traffic scenes to clusters. Due to the huge amount of available data, i.e. recorded traffic scenes, an automatic approach for grouping is necessary for a comprehensive analysis. This leads to Research goal 3.

> *How to automatically cluster traffic scenes in a data-driven manner in terms of their similarity?*

Here, the challenge of determining the similarity of traffic scenes is solved by a self-supervised contrastive approach. Geometric augmentations are performed on a traffic scene. The resulting deviations in the graph description are used to position similar samples close together in the embedding space. Within the embedding space structured in this way, clusters of similar traffic scenes are formed by focusing on their vehicle constellation.

The next two research objectives are closely linked. The structured scene space created by clustering contains a collection of scenes. In order to make testing efficient later on, these must be further enriched with key parameters. This leads to the next research question (Research goal 4):

> *How can a traffic scene be evaluated in a generalizable way to indicate its relevance to the test of a Highly Automated Driving Function?*

In the last part of the thesis, an extensive analysis of the initial conditions of traffic scenes (so called seed-scenes) is therefore carried out. Based on many simulation runs with different behavior models, many possible futures that can arise from the initial condition are generated. An important challenge here are the behavior models of the traffic participants (Research goal 5).

> *How does the behavior of dynamic traffic participants have to be defined in the simulation to accurately represent the scenarios for testing?*

In the section on behavior modeling, two approaches to trajectory prediction based on imitation learning are presented. Both models are based on the graphical scene description from the first part of the thesis, which allows explicit interactions between traffic participants to be fed into the model. Due to the real data-driven approach, both models are able to generate realistic trajectories and corresponding high fidelity in the context of the respective traffic situation. In order to evaluate the various simulated child-scenarios resulting from the different combinations of behavior models, criticality metrics are utilized. Six distinct metrics and two aggregation functions are employed, wherein each scenario is characterized by 12 unique scalars. The various metrics provide a comprehensive overview of the different aspects of the respective simulated scenario. A subsequent analysis of the distribution of all resulting simulation runs of a seed-scenes offers insight into its potential for a critical situation to arise. Through these generic criticality metrics and a large number of simulation runs with different behaviors, the traffic scenes are evaluated as independently as possible. This can improve the transferability of this method for a wide range of Highly Automated Driving Functions. The information about criticality is integrated with the clusters created in the second part, resulting in clusters of similar traffic scenes that are, for example, particularly critical, particularly uncritical, or that highlight a particular aspect of a scene.

8.2 Discussion

All-encompassing representation of reality

In general, each model covers only a specific part that is particularly important for the problem under consideration, and cannot represent reality in its entirety. This is an important limitation that becomes relevant when describing the scenes using the SSG model. Since the SSG is designed in such a way that road geometries are intentionally abstracted, irregularities can no longer be depicted in this context. An example of this is be a traffic participant acting outside the regular road. This problem can be extended to the behavior of traffic participants who do not follow the rules of the road. For example, a prohibited U-turn is not considered as a possibility for a relation within the graph. Although a temporal analysis of the SSG of such a scenario would provide information about a rule violation due to an abrupt change of relations, this is not taken into account for a potential future in a static scene. This challenge of comprehensive coverage, or all so-called long-tail events, can also be seen in the modeling of behavior. Particularly in the case of learned behavior models, it can be observed that they imitate existing movement sequences in the context of already seen situations, as specified by the training paradigm. Atypical behavior can be caused to a certain extent by parameterization, but only covers a small part of possible dynamic anomalies. Furthermore, the resimulation is limited to six different types of behavior, mainly for runtime reasons. This covers a broad spectrum of behaviors, but clearly not all behaviors that occur in reality. The present study focuses primarily on providing an initial estimation of traffic scenes. A more in-depth analysis that takes more long-tail events into account falls outside the scope of this work and can be considered as a follow-up step (see Section 8.3).

Similarity of traffic scenes
Comparing and classifying traffic scenes or traffic scenarios remains a challenge. One problem here is the lack of a metric that allows a full statement to be made about the similarity of two traffic scenes or traffic scenarios. Most related works attempt to relate time series of directly measurable attributes, such as trajectories, to each other. Although the dynamics of traffic participants is a key feature, the independent consideration of individual trajectories hardly takes into account highly interactive traffic situations and the resulting interaction in the metric. A scenario catalog or a manually labeled benchmark is only partially able to solve this problem. Although individual scenarios can be precisely defined in this way, these approaches quickly reach their limits in traffic scenes with many traffic participants. An atomic approach, i.e. considering each situation of individual traffic participants separately, can provide a workaround here. However, there is a risk that this distorts the overall impression of the scene or scenario. The approach presented in this work attempts to circumvent this problem by mapping individual relationships and then aggregating them using a learned procedure. As a result, attributes are also mapped across the entire scene. The problem that arises here is traceability. Because the neural network architecture is a kind of black box, it is not always clear, why some scenes have been merged into a cluster. This aspect of grouping, which is not monitored by humans, makes it difficult for humans to understand the groups or put them into words, but a semantic description of the clusters is not necessarily required for later testing in order to be able to use them.

8.3 Outlook

This dissertation describes a framework for the classification and the generic evaluation of traffic scenes. This framework can be used for the automatic grouping of recordings. The behavior models used in the resimulation represent a wide range of behaviors, but extreme corner cases are not explicitly evoked. This is where methods, that specialize in finding extreme situations, can come into play. For example, the traffic scenes identified in this work can be used as a starting point for scenario exploration methods. Within the scenario exploration approach, the time course of all traffic participants can then be varied until the most critical situations possible arise for a selected ego-vehicle. In the exploration approach, a driving function is used as the SuT, whereby the behavior of all actors can be specifically optimized for it.

One challenge of the approach shown in this dissertation is its real-time applicability. An important part of this framework is the simulation, which requires a significant amount of time. A follow-up approach can use the criticality potential generated in this thesis as a label for a machine-learned approach for direct classification. This end-to-end learned framework can be used to implement a very time-efficient evaluation of a traffic scene. However, a particular focus would have to be on the comprehensibility and parameterization of the machine learning processes.

List of Figures

List of Tables

Own Publications

Conference Paper

This index lists all publications in which the author of this dissertation is either the first author or has contributed to the publication as a co-author (in the form of problem definition, solution, discussion or experimental evaluation).

[1] Maximilian Zipfl, Tobias Fleck, Marc Rene Zofka, and J. Marius Zöllner. From Traffic Sensor Data To Semantic Traffic Descriptions: The Test Area Autonomous Driving Baden-Württemberg Dataset (TAF-BW Dataset). In *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, pages 1–7, Rhodes, Greece, September 2020. IEEE.

[2] Maximilian Zipfl, Nina Koch, and J. Marius Zöllner. A Comprehensive Review on Ontologies for Scenario-based Testing in the Context of Autonomous Driving. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–7, Anchorage, AK, USA, June 2023. IEEE.

[3] Maximilian Zipfl, Barbara Schütt, and J. Marius Zöllner. Scene-Extrapolation: Generating Interactive Traffic Scenarios. In *2024 IEEE Intelligent Vehicles Symposium (IV)*, pages 2483–2490, Jeju Island, Republic of Korea, June 2024. IEEE.

[4] Maximilian Zipfl, Moritz Jarosch, and J. Marius Zöllner. Traffic Scene Similarity: a Graph-based Contrastive Learning Approach. In *2023 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 221–227, Mexico City, Mexico, December 2023. IEEE.

[5] Maximilian Zipfl, Sven Spickermann, and J. Marius Zöllner. Utilizing Hybrid Trajectory Prediction Models to Recognize Highly Interactive Traffic Scenarios. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 799–805, Bilbao, Spain, September 2023. IEEE.

[6] Maximilian Zipfl, Barbara Schütt, J. Marius Zöllner, and Eric Sax. Fingerprint of a Traffic Scene: an Approach for a Generic and Independent Scene Assessment. In *2022 International Conference on Electrical, Computer,*

Communications and Mechatronics Engineering (ICECCME), pages 1–8, Maldives, Maldives, November 2022. IEEE.

[7] Maximilian Zipfl, Felix Hertlein, Achim Rettinger, Steffen Thoma, Lavdim Halilaj, Juergen Luettin, Stefan Schmid, and Cory Henson. Relation-based Motion Prediction using Traffic Scene Graphs. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 825–831, Macau, China, October 2022. IEEE.

[8] Maximilian Zipfl and J. Marius Zöllner. Towards Traffic Scene Description: The Semantic Scene Graph. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3748–3755, Macau, China, October 2022. IEEE.

[9] Maximilian Zipfl, Moritz Jarosch, and J. Marius Zöllner. Self Supervised Clustering of Traffic Scenes using Graph Representations. In *2022 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME)*, pages 1–7, Maldives, Maldives, November 2022. IEEE.

[10] Barbara Schütt, Maximilian Zipfl, J. Marius Zöllner, and Eric Sax. Inverse Universal Traffic Quality - a Criticality Metric for Crowded Urban Traffic Scenes. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–7, Anchorage, AK, USA, June 2023. IEEE.

[11] Felix Naumann, Alexander andr Hertlein, Daniel Grimm, Maximilian Zipfl, Steffen Thoma, Achim Rettinger, Lavdim Halilaj, Juergen Luettin, Stefan Schmid, and Holger Caesar. Lanelet2 for nuScenes: Enabling Spatial Semantic Relationships and Diverse Map-based Anchor Paths. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 3248–3257, Vancouver, BC, Canada, June 2023. IEEE.

[12] Lars Töttel, Maximilian Zipfl, Daniel Bogdoll, Marc René Zofka, and J. Marius Zöllner. Reliving the Dataset: Combining the Visualization of Road Users' Interactions with Scenario Reconstruction in Virtual Reality. In *2021 6th International Conference on Intelligent Transportation Engineering (ICITE 2021)*, volume 901. Springer Nature Singapore, Singapore, 2022.

[13] Marc Rene Zofka, Lars Töttel, Maximilian Zipfl, Marc Heinrich, Tobias Fleck, Patrick Schulz, and J. Marius Zöllner. Pushing ROS towards the Dark Side: A ROS-based Co-Simulation Architecture for Mixed-Reality Test Systems for Autonomous Vehicles. In *2020 IEEE International Conference on Multisensor Fusion and Integration for Intelligent Systems (MFI)*, pages 204–211, Karlsruhe, Germany, September 2020. IEEE.

[14] Daniel Grimm, Maximilian Zipfl, Felix Hertlein, Alexander Naumann, Juergen Luettin, Steffen Thoma, Stefan Schmid, Lavdim Halilaj, Achim Rettinger, and J. Marius Zöllner. Heterogeneous Graph-based Trajectory Prediction using Local Map Context and Social Interactions. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 2901–2907, Bilbao, Spain, September 2023. IEEE.

[15] Tobias Fleck, Maximilian Zipfl, and J. Marius Zöllner. Semi-Automatic Ground Truth Trajectory Estimation and Smoothing using Roadside Cameras. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 4577–4583, Bilbao, Spain, September 2023. IEEE.

[16] Sven Ochs, Jens Doll, Daniel Grimm, Tobias Fleck, Marc Heinrich, Stefan Orf, Albert Schotschneider, Helen Gremmelmaier, Rupert Polley, Svetlana Pavlitska, Maximilian Zipfl, Helen Schneider, Ferdinand Mütsch, Daniel Bogdoll, Florian Kuhnt, Philip Schörner, Marc René Zofka, and J. Marius Zöllner. One Stack to Rule them All: To Drive Automated Vehicles, and Reach for the 4th level, April 2024. arXiv:2404.02645 [cs].

[17] Marc Heinrich, Maximilian Zipfl, Marc Uecker, Sven Ochs, Martin Gontscharow, Tobias Fleck, Jens Doll, Philip Schörner, Christian Hubschneider, Marc René Zofka, Alexander Viehl, and J. Marius Zöllner. CoCar NextGen: a Multi-Purpose Platform for Connected Autonomous Driving Research, April 2024. arXiv:2404.17550 [cs].

[18] Sven Ochs, Melih Yazgan, Rupert Polley, Albert Schotschneider, Stefan Orf, Marc Uecker, Maximilian Zipfl, Julian Burger, Abhishek Vivekanandan, Jennifer Amritzer, Marc René Zofka, and J. Marius Zöllner. Empowering Autonomous Shuttles with Next-Generation Infrastructure, October 2024. arXiv:2410.20989 [cs].

Patents

[19] Maximilian Zipfl, Achim Rettinger, Cory Henson, Felix Hertlein, Lavdim Halilaj, Stefan Schmid, and Steffen Thoma. Predicting the behavior of road users based on graph representation of a traffic situation, March 2023. Publication number: US-20240071211-A1.

Student work

This index lists student theses that the author of this work has supervised as part of his research.

[20] Moritz Jarosch. Clustering of Traffic Scene Graphs using Variational Autoencoders. Master's thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe Germany, January 2022.

[21] Benedikt Kaas. Generation of Traffic Scenes from Abstract Traffic Descriptions. Bachelor's Thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe Germany, January 2023.

[22] Nina Koch. Scenario-Based Validation for Autonomous Driving: Matching and Integration of Ontology-Based Scenario Modelling Approaches. Master's thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe Germany, August 2021.

[23] Xuanheng Liu. Simulation-based Scenario Analysis for Testing Autonomous Driving Functions. Forschungsarbeit, Universität Stuttgart, Stuttgart Deutschland, September 2023.

[24] Marc Nikolaus. Analyse von Verhaltensmodellen bezüglich Verkehrsszenen mithilfe von Graph-Neuronalen-Netzen. Bachelor's Thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe Germany, July 2021.

[25] Sven Spickermann. Nachahmung menschlichen Verkehrsverhaltens durch Anwendung einer Kombination aus CNN und GNN. Master's thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe Germany, August 2022.

[26] Feng Zhou. Traffic Motion Prediction based on a Graph Representation with Spatial-Temporal Graph Neural Networks. Master's thesis, Karlsruhe Institute of Technology (KIT), Karlsruhe Germany, November 2021.

Bibliography

[27] Zhanxiang Chai, Tianxin Nie, and Jan Becker. *Autonomous Driving Changes the Future*. Springer Singapore, Singapore, 2021.

[28] Elisabeth Shi, Tom Michael Gasser, Andre Seeck, and Rico Auerswald. The Principles of Operation Framework: A Comprehensive Classification Concept for Automated Driving Functions. *SAE International Journal of Connected and Automated Vehicles*, 3(1):12–03–01–0003, February 2020.

[29] Waymo. Self-Driving Car Technology for a Reliable Ride - Waymo Driver. `https://waymo.com/waymo-driver/`, April 2024. Accessed on 24.04.2024.

[30] IEEE Spectrum. What Full Autonomy Means for the Waymo Driver - IEEE Spectrum. `https://spectrum.ieee.org/full-autonomy-waymo-driver`, March 2021. Accessed on 25.04.2024.

[31] Mary L. Cummings. What Self-Driving Cars Tell Us About AI Risks. `https://spectrum.ieee.org/self-driving-cars-2662494269`, July 2023. Accessed on 25.04.2024.

[32] Wolfgang Rudschies. Autonomes Fahren: So fahren wir in Zukunft, October 2022.

[33] Erwin De Gelder, Jan-Pieter Paardekooper, Arash Khabbaz Saberi, Hala Elrofai, Olaf Op Den Camp, Steven Kraines, Jeroen Ploeg, and Bart De Schutter. Towards an Ontology for Scenario Definition for the Assessment of Automated Vehicles: An Object-Oriented Framework. *IEEE Transactions on Intelligent Vehicles*, 7(2):300–314, June 2022.

[34] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. On a Formal Model of Safe and Scalable Self-driving Cars, October 2018. arXiv:1708.06374 [cs, stat].

[35] Walther Wachenfeld and Hermann Winner. The Release of Autonomous Vehicles. In J. Christian Gerdes, Markus Maurer, Barbara Lenz, and Hermann Winner, editors, *Autonomous Driving*, pages 425–449. Springer Berlin Heidelberg, Berlin, Heidelberg, 2016.

[36] Alexander Pretschner, Florian Hauer, and Tabea Schmidt. Tests für automatisierte und autonome Fahrsysteme: Wiederverwendung aufgezeichneter Fahrten ist nicht zu rechtfertigen. *Informatik Spektrum*, 44(3):214–218, June 2021.

[37] Sarah Ryglewski. Verordnung zur Regelung des Betriebs von Kraftfahrzeugen mit automatisierter und autonomer Fahrfunktion und zur Änderung straßenverkersrechtlicher Vorschriften, February 2022.

[38] European Commission. Commission Implementing Regulation (EU) 2022/1426 of 5 August 2022 laying down rules for the application of Regulation (EU) 2019/2144 of the European Parliament and of the Council as regards uniform procedures and technical specifications for the type-approval of the automated driving system (ADS) of fully automated vehicles, August 2022.

[39] Knut Ringat, André Kavai, and Markus Huber. The Project - KIRA - Autonomes Fahren. https://kira-autonom.de/, March 2024. Accessed on 23.07.2024.

[40] Eric Thorn, Shawn Kimmel, and Michelle Chaka. A Framework for Automated Driving System Testable Cases and Scenarios. Final Report DOT HS 812 623, National Highway Traffic Safety Administration, September 2018.

[41] N. Webb, D. Smith, C. Ludwick, T. W. Victor, Q. Hommes, F. Favorò, G. Ivanov, and T Daniel. Waymo-Safety-Methodologies-and-Readiness-Determinations, 2020.

[42] Till Menzel, Gerrit Bagschik, and Markus Maurer. Scenarios for Development, Test and Validation of Automated Vehicles, April 2018. arXiv:1801.08598 [cs].

[43] Jens Mazzega. Pegasus Abschlussbericht, February 2020.

[44] Lukas Birkemeyer, Tobias Pett, Andreas Vogelsang, Christoph Seidl, and Ina Schaefer. Feature-Interaction Sampling for Scenario-based Testing of Advanced Driver Assistance Systems. In *Proceedings of the 16th International Working Conference on Variability Modelling of Software-Intensive Systems*, pages 1–10, Florence Italy, February 2022. ACM.

[45] Barbara Schütt, Marc Heinrich, Sonja Marahrens, J. Marius Zöllner, and Eric Sax. An Application of Scenario Exploration to Find New Scenarios for the Development and Testing of Automated Driving Systems in Urban Scenarios. In *Proceedings of the 8th International Conference on Vehicle Technology and Intelligent Transport Systems*, pages 338–345, 2022. arXiv:2205.08202 [cs].

[46] Rupak Majumdar, Aman Mathur, Marcus Pirron, Laura Stegner, and Damien Zufferey. Paracosm: A Test Framework for Autonomous Driving Simulations. In Esther Guerra and Mariëlle Stoelinga, editors, *Fundamental Approaches to Software Engineering*, volume 12649, pages 172–195. Springer International Publishing, Cham, 2021. Series Title: Lecture Notes in Computer Science.

[47] Wenhao Ding, Chejian Xu, Mansur Arief, Haohong Lin, Bo Li, and Ding Zhao. A Survey on Safety-Critical Driving Scenario Generation – A Methodological Perspective, June 2023. arXiv:2202.02215 [cs].

[48] Lukas Birkemeyer, Christian King, and Ina Schaefer. Is Scenario Generation Ready for SOTIF? A Systematic Literature Review. In *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, pages 472–479, Bilbao, Spain, September 2023. IEEE.

[49] Hardi Hungar. A Concept of Scenario Space Exploration with Criticality Coverage Guarantees: Extended Abstract. In Tiziana Margaria and Bernhard Steffen, editors, *Leveraging Applications of Formal Methods, Verification and Validation: Applications*, volume 12478, pages 293–306. Springer International Publishing, Cham, 2020. Series Title: Lecture Notes in Computer Science.

[50] Holger Caesar, Juraj Kabzan, Kok Seang Tan, Whye Kit Fong, Eric Wolff, Alex Lang, Luke Fletcher, Oscar Beijbom, and Sammy Omari. nuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles, February 2022. arXiv:2106.11810 [cs].

[51] Florian Hauer, Tabea Schmidt, Bernd Holzmuller, and Alexander Pretschner. Did We Test All Scenarios for Automated and Autonomous Driving Systems? In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2950–2955, Auckland, New Zealand, October 2019. IEEE.

[52] Shuhan Tan, Kelvin Wong, Shenlong Wang, Sivabalan Manivasagam, Mengye Ren, and Raquel Urtasun. SceneGen: Learning to Generate Realistic Traffic Scenes, January 2021. arXiv:2101.06541 [cs].

[53] Matthias Schreier. Environment representations for automated on-road vehicles. *at - Automatisierungstechnik*, 66(2):107–118, February 2018.

[54] Florian Hauer, Ilias Gerostathopoulos, Tabea Schmidt, and Alexander Pretschner. Clustering Traffic Scenarios Using Mental Models as Little as Possible. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1007–1012, Las Vegas, NV, USA, October 2020. IEEE.

[55] Jonas Kerber, Sebastian Wagner, Korbinian Groh, Dominik Notz, Thomas Kuhbeck, Daniel Watzenig, and Alois Knoll. Clustering of the Scenario Space for the Assessment of Automated Driving. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 578–583, Las Vegas, NV, USA, October 2020. IEEE.

[56] Nick Harmening, Marin Biloš, and Stephan Günnemann. Deep Representation Learning and Clustering of Traffic Scenarios, July 2020. arXiv:2007.07740 [cs, stat].

[57] Lennart Ries, Philipp Rigoll, Thilo Braun, Thomas Schulik, Johannes Daube, and Eric Sax. Trajectory-Based Clustering of Real-World Urban Driving Sequences with Multiple Traffic Objects. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1251–1258, Indianapolis, IN, USA, September 2021. IEEE.

[58] Halil Beglerovic, Steffen Metzner, and Martin Horn. Challenges for the Validation and Testing of Automated Driving Functions. In Carolin Zachäus, Beate Müller, and Gereon Meyer, editors, *Advanced Microsystems for Automotive Applications 2017*, pages 179–187. Springer International Publishing, Cham, 2018. Series Title: Lecture Notes in Mobility.

[59] Lukas Westhofen, Christian Neurohr, Tjark Koopmann, Martin Butz, Barbara Schütt, Fabian Utesch, Birte Neurohr, Christian Gutenkunst, and Eckard Böde. Criticality Metrics for Automated Driving: A Review and Suitability Analysis of the State of the Art. *Archives of Computational Methods in Engineering*, 30(1):1–35, January 2023.

[60] Julian Bernhard, Klemens Esterle, Patrick Hart, and Tobias Kessler. BARK: Open Behavior Benchmarking in Multi-Agent Environments. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 6201–6208, Las Vegas, NV, USA, October 2020. IEEE.

[61] ISO. *ISO:21448:2022, Road vehicles. Safety of the intended functionality*. British Standards Institution, London, 2022. OCLC: 1351679387.

[62] ISO. *ISO:26262-2:2018, Road vehicles - Functional safety*, volume 2. British Standards Institution, London, 2018.

[63] Lars Schnieder and René S. Hosse. *Leitfaden Safety of the Intended Functionality: Verfeinerung der Sicherheit der Sollfunktion auf dem Weg zum autonomen Fahren*. essentials. Springer Fachmedien Wiesbaden, Wiesbaden, 2019.

[64] Nidhi Kalra and Susan M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? *Transportation Research Part A: Policy and Practice*, 94:182–193, December 2016.

[65] Stefan Riedmaier, Thomas Ponn, Dieter Ludwig, Bernhard Schick, and Frank Diermeyer. Survey on Scenario-Based Safety Assessment of Automated Vehicles. *IEEE Access*, 8:87456–87477, 2020.

[66] State of California. Autonomous Vehicle Permit Holders Report 5.7 Million Test Miles in California. https://www.dmv.ca.gov/portal/news-and-media/autonomous-vehicle-permit-holders-report-5-7-million-test-miles-in-california/, October 2023. Accessed on 23.10.2023.

[67] Laura Fraade-Blanar, Marjory Blumenthal, James Anderson, and Nidhi Kalra. *Measuring Automated Vehicle Safety: Forging a Framework*. RAND Corporation, 2018.

[68] David Nistér, Hon-Leung Lee, Julia Ng, and Yishou Wang. An Introduction to the Safety Force Field (Nvidia), 2019.

[69] Tong Zhao, Ekim Yurtsever, Joel A. Paulson, and Giorgio Rizzoni. Formal Certification Methods for Automated Vehicle Safety Assessment. *IEEE Transactions on Intelligent Vehicles*, 8(1):232–249, January 2023.

[70] Fabian Schuldt. *Ein Beitrag für den methodischen Test von automatisierten Fahrfunktionen mit Hilfe von virtuellen Umgebungen*. PhD thesis, Universitätsbibliothek Braunschweig, April 2017.

[71] Gerrit Bagschik, Till Menzel, and Markus Maurer. Ontology based Scene Creation for the Development of Automated Vehicles, April 2018. arXiv: 1704.01006 [cs].

[72] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving. In *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, pages 982–988, Gran Canaria, Spain, September 2015. IEEE.

[73] Jian Sun, He Zhang, Huajun Zhou, Rongjie Yu, and Ye Tian. Scenario-Based Test Automation for Highly Automated Vehicles: A Review and Paving the Way for Systematic Safety Assurance. *IEEE Transactions on Intelligent Transportation Systems*, 23(9):14088–14103, September 2022.

[74] Franz Wotawa, Josip Bozic, and Yihao Li. Ontology-based Testing: An Emerging Paradigm for Modeling and Testing Systems and Software. In *2020 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*, pages 14–17, Porto, Portugal, October 2020. IEEE.

[75] Florian Klueck, Yihao Li, Mihai Nica, Jianbo Tao, and Franz Wotawa. Using Ontologies for Test Suites Generation for Automated and Autonomous Driving Functions. In *2018 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, pages 118–123, Memphis, TN, USA, October 2018. IEEE.

[76] Jinkang Cai, Weiwen Deng, Haoran Guang, Ying Wang, Jiangkun Li, and Juan Ding. A Survey on Data-Driven Scenario Generation for Automated Vehicle Testing. *Machines*, 10(11):1101, November 2022. Number: 11 Publisher: Multidisciplinary Digital Publishing Institute.

[77] Jacob Langner, Johannes Bach, Lennart Ries, Stefan Otten, Marc Holzapfel, and Eric Sax. Estimating the Uniqueness of Test Scenarios derived from Recorded Real-World-Driving-Data using Autoencoders. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1860–1866, Changshu, China, June 2018. IEEE.

[78] Wenhao Ding, Baiming Chen, Minjun Xu, and Ding Zhao. Learning to Collide: An Adaptive Safety-Critical Scenarios Generating Method. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 2243–2250, Las Vegas, NV, USA, October 2020. IEEE.

[79] Jingkang Wang, Ava Pun, James Tu, Sivabalan Manivasagam, Abbas Sadat, Sergio Casas, Mengye Ren, and Raquel Urtasun. AdvSim: Generating Safety-Critical Scenarios for Self-Driving Vehicles. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9904–9913, Nashville, TN, USA, June 2021. IEEE.

[80] Shuo Feng, Xintao Yan, Haowei Sun, Yiheng Feng, and Henry X. Liu. Intelligent driving intelligence test for autonomous vehicles with naturalistic and adversarial environment. *Nature Communications*, 12(1):748, February 2021.

[81] William L Hamilton. *Graph Representation Learning*, volume 14 of *Synthesis Lectures on Artificial Intelligence and Machine Learning*. Morgan and Claypoo, 1 edition, 2022.

[82] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. Attention Is All You Need, December 2017. arXiv:1706.03762 [cs].

[83] Vijay Prakash Dwivedi and Xavier Bresson. A Generalization of Transformer Networks to Graphs, January 2021. arXiv:2012.09699 [cs].

[84] A Geiger, P Lenz, C Stiller, and R Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, September 2013.

[85] Wei Zhan, Liting Sun, Di Wang, Haojie Shi, Aubrey Clausse, Maximilian Naumann, Julius Kummerle, Hendrik Konigshof, Christoph Stiller, Arnaud de La Fortelle, and Masayoshi Tomizuka. INTERACTION Dataset: An INTERnational, Adversarial and Cooperative moTION Dataset in Interactive Driving Scenarios with Semantic Maps, September 2019. arXiv:1910.03088 [cs, eess].

[86] Julian Bock, Robert Krajewski, Tobias Moers, Steffen Runde, Lennart Vater, and Lutz Eckstein. The inD Dataset: A Drone Dataset of Naturalistic Road User Trajectories at German Intersections. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 1929–1934, Las Vegas, NV, USA, October 2020. IEEE.

[87] Robert Krajewski, Julian Bock, Laurent Kloeker, and Lutz Eckstein. 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)*, pages 2118–2125, Maui, HI, USA, November 2018. IEEE.

[88] Pengchuan Xiao, Zhenlei Shao, Steven Hao, Zishuo Zhang, Xiaolin Chai, Judy Jiao, Zesong Li, Jian Wu, Kai Sun, Kun Jiang, Yunlong Wang, and Diange Yang. PandaSet: Advanced Sensor Suite Dataset for Autonomous Driving, December 2021. arXiv:2112.12610 [cs].

[89] Holger Caesar, Varun Bankiti, Alex H. Lang, Sourabh Vora, Venice Erin Liong, Qiang Xu, Anush Krishnan, Yu Pan, Giancarlo Baldan, and Oscar Beijbom. nuScenes: A multimodal dataset for autonomous driving, May 2020. arXiv:1903.11027 [cs, stat].

[90] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst, December 2018. arXiv:1812.03079 [cs].

[91] Jianyu Chen, Bodi Yuan, and Masayoshi Tomizuka. Deep Imitation Learning for Autonomous Driving in Generic Urban Scenarios with Enhanced Safety, October 2019. arXiv:1903.00640 [cs].

[92] Wenyuan Zeng, Shenlong Wang, Renjie Liao, Yun Chen, Bin Yang, and Raquel Urtasun. DSDNet: Deep Structured self-Driving Network, August 2020. arXiv:2008.06041 [cs].

[93] Rudi Studer, V.Richard Benjamins, and Dieter Fensel. Knowledge engineering: Principles and methods. *Data & Knowledge Engineering*, 25(1-2):161–197, March 1998.

[94] Grega Jakus, Veljko Milutinović, Sanida Omerović, and Sašo Tomažič. *Concepts, Ontologies, and Knowledge Representation*. SpringerBriefs in Computer Science. Springer New York, New York, NY, USA, 2013.

[95] Yanhui Lv. An approach to ontologies integration. In *2011 Eighth International Conference on Fuzzy Systems and Knowledge Discovery (FSKD)*, pages 1262–1266, Shanghai, China, July 2011. IEEE.

[96] Dieter Fensel. From Web to Semantic Web. In Mick Kerrigan and Michal Zaremba, editors, *Implementing Semantic Web Services*, pages 3–25. Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2008.

[97] Wei Chen and Leïla Kloul. An Ontology-based Approach to Generate the Advanced Driver Assistance Use Cases of Highway Traffic:. In *Proceedings of the 10th International Joint Conference on Knowledge Discovery, Knowledge Engineering and Knowledge Management*, pages 75–83, Seville, Spain, 2018. SCITEPRESS - Science and Technology Publications.

[98] Christopher Medrano-Berumen and Mustafaff Ilhan Akbas. Abstract Simulation Scenario Generation for Autonomous Vehicle Verification. In *2019 SoutheastCon*, pages 1–6, Huntsville, AL, USA, April 2019. IEEE.

[99] Simon Ulbrich, Tobias Nothdurft, Markus Maurer, and Peter Hecker. Graph-based context representation, environment modeling and information aggregation for automated driving. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 541–547, MI, USA, June 2014. IEEE.

[100] Michael Hülsen, J. Marius Zöllner, and Christian Weiss. Traffic intersection situation description ontology for advanced driver assistance. In *2011 IEEE Intelligent Vehicles Symposium (IV)*, pages 993–999, Baden-Baden, Germany, June 2011. IEEE.

[101] Martin Buechel, Gereon Hinz, Frederik Ruehl, Hans Schroth, Csaba Gyoeri, and Alois Knoll. Ontology-based traffic scene modeling, traffic regulations dependent situational awareness and decision-making for automated vehicles. In *2017 IEEE Intelligent Vehicles Symposium (IV)*, pages 1471–1476, Los Angeles, CA, USA, June 2017. IEEE.

[102] Ralf Regele. Using Ontology-Based Traffic Models for More Efficient Decision Making of Autonomous Vehicles. In *Fourth International Conference on Autonomic and Autonomous Systems (ICAS'08)*, pages 94–99, Gosier, Guadeloupe, March 2008. IEEE.

[103] Fang Fang, Shotaro Yamaguchi, and Abdelaziz Khiat. Ontology-based Reasoning Approach for Long-term Behavior Prediction of Road Users. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2068–2073, Auckland, New Zealand, October 2019. IEEE.

[104] Wei Chen and Leïla Kloul. An Advanced Driver Assistance Test Cases Generation Methodology Based on Highway Traffic Situation Description Ontologies. In Ana Fred, Ana Salgado, David Aveiro, Jan Dietz, Jorge Bernardino, and Joaquim Filipe, editors, *Knowledge Discovery, Knowledge Engineering and Knowledge Management*, volume 1222, pages 93–113. Springer International Publishing, Cham, 2020. Series Title: Communications in Computer and Information Science.

[105] M. Elgharbawy, A. Schwarzhaupt, M. Frey, and F. Gauterin. Ontology-based adaptive testing for automated driving functions using data mining techniques. *Transportation Research Part F: Traffic Psychology and Behaviour*, 66:234–251, October 2019.

[106] Martin Herrmann, Christian Witt, Laureen Lake, Stefani Guneshka, Christian Heinzemann, Frank Bonarens, Patrick Feifel, and Simon Funke. Using ontologies for dataset engineering in automotive AI applications. In *2022 Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pages 526–531, Antwerp, Belgium, March 2022. IEEE.

[107] Daniel Bogdoll, Stefani Guneshka, and J. Marius Zöllner. One Ontology to Rule Them All: Corner Case Scenarios for Autonomous Driving, October 2022. arXiv:2209.00342 [cs].

[108] ASAM e.V. ASAM OpenScenario. https://www.asam.net/standards/detail/openscenario/, Mar 2021. Accessed on 04.02.2024.

[109] Steffen Staab and Rudi Studer, editors. *Handbook on Ontologies*. Springer Berlin Heidelberg, Berlin, Heidelberg, Germany, 2009.

[110] Lu Huang, Huawei Liang, Biao Yu, Bichun Li, and Hui Zhu. Ontology-Based Driving Scene Modeling, Situation Assessment and Decision Making for Autonomous Vehicles. In *2019 4th Asia-Pacific Conference on Intelligent Robot Systems (ACIRS)*, pages 57–62, Nagoya, Japan, July 2019. IEEE.

[111] Ralf Kohlhaas, Thomas Bittner, Thomas Schamm, and J. Marius Zollner. Semantic state space for high-level maneuver planning in structured traffic scenes. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*, pages 1060–1065, Qingdao, China, October 2014. IEEE.

[112] Dominik Petrich, Darius Azarfar, Florian Kuhnt, and J. Marius Zollner. The Fingerprint of a Traffic Situation: A Semantic Relationship Tensor for Situation Description and Awareness. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 429–435, Maui, HI, USA, November 2018. IEEE.

[113] Lihua Zhao, Ryutaro Ichise, Zheng Liu, Seiichi Mita, and Yutaka Sasaki. Ontology-Based Driving Decision Making: A Feasibility Study at Uncontrolled Intersections. *IEICE Transactions on Information and Systems*, E100.D(7):1425–1439, 2017.

[114] Lukas Westhofen, Christian Neurohr, Martin Butz, Maike Scholtes, and Michael Schuldes. Using Ontologies for the Formalization and Recognition of Criticality for Automated Driving. *IEEE Open Journal of Intelligent Transportation Systems*, 3:519–538, 2022.

[115] Hao Zhou, Dongchun Ren, Huaxia Xia, Mingyu Fan, Xu Yang, and Hai Huang. AST-GNN: An attention-based spatio-temporal graph neural network for Interaction-aware pedestrian trajectory prediction. *Neurocomputing*, 445:298–308, July 2021.

[116] Cunjun Yu, Xiao Ma, Jiawei Ren, Haiyu Zhao, and Shuai Yi. Spatio-Temporal Graph Transformer Networks for Pedestrian Trajectory Prediction, July 2020. arXiv:2005.08514 [cs].

[117] Sirin Haddad, Meiqing Wu, He Wei, and Siew Kei Lam. Situation-Aware Pedestrian Trajectory Prediction with Spatio-Temporal Attention Model. *24th Computer Vision Winter Workshop*, 2019. arXiv:1902.05437 [cs].

[118] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. GRIP: Graph-based Interaction-aware Trajectory Prediction. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 3960–3966, Auckland, New Zealand, October 2019. IEEE.

[119] Xin Li, Xiaowen Ying, and Mooi Choo Chuah. GRIP++: Enhanced Graph-based Interaction-aware Trajectory Prediction for Autonomous Driving, May 2020. arXiv:1907.07792 [cs].

[120] Jiyang Gao, Chen Sun, Hang Zhao, Yi Shen, Dragomir Anguelov, Congcong Li, and Cordelia Schmid. VectorNet: Encoding HD Maps and Agent Dynamics from Vectorized Representation, May 2020. arXiv: 2005.04259.

[121] Frederik Diehl, Thomas Brunner, Michael Truong Le, and Alois Knoll. Graph Neural Networks for Modelling Traffic Participant Interaction.

In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 695–701, Paris, France, June 2019. IEEE.

[122] Hengbo Ma, Yaofeng Sun, Jiachen Li, and Masayoshi Tomizuka. Multi-Agent Driving Behavior Prediction across Different Scenarios with Self-Supervised Domain Knowledge. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 3122–3129, Indianapolis, IN, USA, September 2021. IEEE.

[123] Wenshuo Wang, Letian Wang, Chengyuan Zhang, Changliu Liu, and Lijun Sun. Social Interactions for Autonomous Driving: A Review and Perspectives. *Foundations and Trends in Robotics*, 10(3-4):198–376, 2022.

[124] Gerald J. S. Wilde. Social Interaction Patterns in Driver Behavior: An Introductory Review. *Human Factors: The Journal of the Human Factors and Ergonomics Society*, 18(5):477–492, October 1976.

[125] G J S. Wilde. Immediate and delayed social interaction in road user behaviour. *Applied Psychology*, 29(4):439–460, October 1980.

[126] Moritz Werling, Julius Ziegler, Sören Kammel, and Sebastian Thrun. Optimal trajectory generation for dynamic street scenarios in a Frenet Frame. In *2010 IEEE International Conference on Robotics and Automation*, pages 987–993, Anchorage, AK, May 2010. IEEE.

[127] Fabian Poggenhans, Jan-Hendrik Pauls, Johannes Janosovits, Stefan Orf, Maximilian Naumann, Florian Kuhnt, and Matthias Mayr. Lanelet2: A high-definition map framework for the future of automated driving. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 1672–1679, Maui, HI, USA, November 2018. IEEE.

[128] Philipp Bender, Julius Ziegler, and Christoph Stiller. Lanelets: Efficient map representation for autonomous driving. In *2014 IEEE Intelligent Vehicles Symposium Proceedings*, pages 420–425, MI, USA, June 2014. IEEE.

[129] Edsger W. Dijkstra. A note on two problems in connexion with graphs. In *Edsger Wybe Dijkstra: his life, work, and legacy*, pages 287–290. Association for Computing Machinery, 2022.

[130] Graphviz Authors. Dot language. `https://graphviz.org/doc/info/lang.html`, Okt 2022. Accessed on 23.07.2024.

[131] Christopher Morris, Nils M. Kriege, Franka Bause, Kristian Kersting, Petra Mutzel, and Marion Neumann. TUDataset: A collection of benchmark datasets for learning with graphs. *CoRR*, abs/2007.08663, 2020. arXiv: 2007.08663 [cs].

[132] Yuanfei Lin and Matthias Althoff. CommonRoad-CriMe: A Toolbox for Criticality Measures of Autonomous Vehicles. In *2023 IEEE Intelligent Vehicles Symposium (IV)*, pages 1–8, Anchorage, AK, USA, June 2023. IEEE.

[133] Barbara Schütt, Markus Steimle, Birte Kramer, Danny Behnecke, and Eric Sax. A taxonomy for quality in simulation-based development and testing of automated driving systems. *IEEE Access*, 10:18631–18644, 2022. arXiv:2102.06588 [cs].

[134] SM Sohel Mahmud, Luis Ferreira, Md Shamsul Hoque, and Ahmad Tavassoli. Application of proximal surrogate indicators for safety evaluation: A review of recent developments and research needs. *IATSS research*, 41(4):153–163, 2017. Publisher: Elsevier.

[135] Philipp Junietz, Jan Schneider, and Hermann Winner. Metrik zur Bewertung der Kritikalität von Verkehrssituationen und -szenarien. In *11. Workshop Fahrerassistenzsysteme und automatisiertes Fahren*, Walting, Germany, March 2017.

[136] John C Hayward. 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, January 1972. Highway Research Board.

[137] Hiroshi Wakabayashi, Yoshihiko Takahashi, Shigehiro Niimi, and Kazumi Renge. Traffic Conflict Analysis using Vehicle Tracking System/Digital VCR and Proposal of a New Conflict Indicator. *INFRASTRUCTURE PLANNING REVIEW*, 20:949–956, 2003.

[138] Walther Wachenfeld, Philipp Junietz, Raphael Wenzel, and Hermann Winner. The worst-time-to-collision metric for situation identification. In *2016 IEEE Intelligent Vehicles Symposium (IV)*, pages 729–734, Gotenburg, Sweden, June 2016. IEEE.

[139] Brian L Allen, B Tom Shin, and Peter J Cooper. Analysis of Traffic Conflicts and Collisions. In *Transportation Research Record*, volume Record 667, pages pp. 67–74., 1978.

[140] Sven Hallerbach. *Simulation-Based Testing of Cooperative and Automated Vehicles*. PhD thesis, Carl von Ossietzky Universität Oldenburg, June 2020.

[141] David Nistér, Hon-Leung Lee, Julia Ng, and Yizhou Wang. The safety force field. *NVIDIA White Paper*, 2019.

[142] Samuel Rosa and Andrea Kocianova. Accuracy of critical gap estimates. *Alexandria Engineering Journal*, 75:565–576, July 2023.

[143] Johannes Bernhard, Mark Schutera, and Eric Sax. Optimizing test-set diversity: Trajectory clustering for scenario-based testing of automated driving systems. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1371–1378, Indianapolis, IN, USA, September 2021. IEEE.

[144] Rui Zhou, Ziqian Lin, Xiangzhi Huang, Jingfeng Peng, and Helai Huang. Testing Scenarios Construction for Connected and Automated Vehicles Based on Dynamic Trajectory Clustering Method. In *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, pages 3304–3308, Macau, China, October 2022. IEEE.

[145] Jonas Wurst, Lakshman Balasubramanian, Michael Botsch, and Wolfgang Utschick. Expert-LaSTS: Expert-Knowledge Guided Latent Space for Traffic Scenarios. In *2022 IEEE Intelligent Vehicles Symposium (IV)*, pages 484–491, Aachen, Germany, June 2022. IEEE.

[146] Christian King, Thilo Braun, Constantin Braess, Jacob Langner, and Eric Sax. Capturing the Variety of Urban Logical Scenarios from Bird-view Trajectories:. In *Proceedings of the 7th International Conference on Vehicle Technology and Intelligent Transport Systems*, pages 471–480, Online Streaming, — Select a Country —, 2021. SCITEPRESS - Science and Technology Publications.

[147] Jiang Bian, Dayong Tian, Yuanyan Tang, and Dacheng Tao. A survey on trajectory clustering analysis, February 2018. arXiv:1802.06971 [cs].

[148] Friedrich Kruber, Jonas Wurst, Eduardo Sanchez Morales, Samarjit Chakraborty, and Michael Botsch. Unsupervised and Supervised Learning with the Random Forest Algorithm for Traffic Scenario Clustering and Classification. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 2463–2470, Paris, France, June 2019. IEEE.

[149] Paulino Cristovao, Hidemoto Nakada, Yusuke Tanimura, and Hideki Asoh. Generating In-Between Images Through Learned Latent Space Representation Using Variational Autoencoders. *IEEE Access*, 8:149456–149467, 2020.

[150] Martin Simonovsky and Nikos Komodakis. GraphVAE: Towards Generation of Small Graphs Using Variational Autoencoders, February 2018. arXiv:1802.03480 [cs].

[151] Thomas N. Kipf and Max Welling. Variational Graph Auto-Encoders, November 2016. arXiv:1611.07308 [cs, stat].

[152] Keyulu Xu, Weihua Hu, Jure Leskovec, and Stefanie Jegelka. How Powerful are Graph Neural Networks?, February 2019. arXiv:1810.00826 [cs, stat].

[153] Yanqiao Zhu, Yuanqi Du, Yinkai Wang, Yichen Xu, Jieyu Zhang, Qiang Liu, and Shu Wu. A Survey on Deep Graph Generation: Methods and Applications, August 2022. arXiv:2203.06714 [cs, q-bio].

[154] Yuning You, Tianlong Chen, Yang Shen, and Zhangyang Wang. Graph Contrastive Learning Automated. In *Proceedings of the 38 th International Conference on Machine Learning, PMLR 139, 2021.*, 2021.

[155] Kaveh Hassani and Amir Hosein Khasahmadi. Contrastive Multi-View Representation Learning on Graphs, June 2020. arXiv:2006.05582 [cs, stat].

[156] Fan-Yun Sun, Jordan Hoffmann, Vikas Verma, and Jian Tang. InfoGraph: Unsupervised and Semi-supervised Graph-Level Representation Learning via Mutual Information Maximization, January 2020. arXiv:1908.01000 [cs, stat].

[157] Zhirong Wu, Yuanjun Xiong, Stella Yu, and Dahua Lin. Unsupervised Feature Learning via Non-Parametric Instance-level Discrimination, May 2018. arXiv:1805.01978 [cs].

[158] Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised Contrastive Learning, March 2021. arXiv:2004.11362 [cs, stat].

[159] Kihyuk Sohn. Improved Deep Metric Learning with Multi-class N-pair Loss Objective. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[160] Carl Doersch, Abhinav Gupta, and Alexei A. Efros. Unsupervised Visual Representation Learning by Context Prediction, January 2016. arXiv:1505.05192 [cs].

[161] Rex Ying, Jiaxuan You, Christopher Morris, Xiang Ren, William L. Hamilton, and Jure Leskovec. Hierarchical Graph Representation Learning with Differentiable Pooling, February 2019. arXiv:1806.08804 [cs, stat].

[162] Robert L. Thorndike. Who belongs in the family? *Psychometrika*, 18(4):267–276, December 1953.

[163] Leland McInnes, John Healy, and James Melville. UMAP: Uniform Manifold Approximation and Projection for Dimension Reduction, September 2020. arXiv:1802.03426 [cs, stat].

[164] Nathan Gartner, C. J. Messer, and A. K. Rathi. Traffic Flow Theory - A State-of-the-Art Report: Revised Monograph on Traffic Flow Theory. *Turner-Fairbank Highway Research Center*, April 2002.

[165] Stefan Kupschick. *Modellierung menschähnlichen Fahrerverhaltens*. PhD thesis, Technische Universität Berlin, Berlin, Deutschland, August 2020.

[166] Denos C. Gazis, Robert Herman, and Richard W. Rothery. Nonlinear Follow-the-Leader Models of Traffic Flow. *Operations Research*, 9(4):545–567, August 1961.

[167] Martin Treiber, Ansgar Hennecke, and Dirk Helbing. Congested traffic states in empirical observations and microscopic simulations. *Physical Review E*, 62(2):1805–1824, August 2000.

[168] Stefan Krauß. *Microscopic Modeling of Traffic Flow: Investigation of Collision Free Vehicle Dynamics*. PhD thesis, Universität zu Köln, April 1998.

[169] Daniel Krajzewicz, Georg Hertkorn, Peter Wagner, and Christian Rössel. SUMO (Simulation of Urban MObility). In *Proceedings of the 4th Middle East Symposium on Simulation and Modelling*, Sharjah (United Arab Emirates), September 2002.

[170] Stéphanie Lefèvre, Dizan Vasquez, and Christian Laugier. A survey on motion prediction and risk assessment for intelligent vehicles. *ROBOMECH Journal*, 1(1):1, December 2014.

[171] Nachiket Deo and Mohan M. Trivedi. Multi-Modal Trajectory Prediction of Surrounding Vehicles with Maneuver based LSTMs. In *2018 IEEE Intelligent Vehicles Symposium (IV)*, pages 1179–1184, June 2018. arXiv:1805.05499 [cs].

[172] Joey Hong, Benjamin Sapp, and James Philbin. Rules of the Road: Predicting Driving Behavior with a Convolutional Model of Semantic Interactions, June 2019. arXiv:1906.08945 [cs].

[173] Philip Schörner, Christian Hubschneider, Jonathan Hartl, Rupert Polley, and J. Marius Zöllner. Grid-Based Micro Traffic Prediction using Fully Convolutional Networks. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 4540–4547, Auckland, New Zealand, October 2019. IEEE.

[174] Yuning Chai, Benjamin Sapp, Mayank Bansal, and Dragomir Anguelov. MultiPath: Multiple Probabilistic Anchor Trajectory Hypotheses for Behavior Prediction, October 2019. arXiv:1910.05449 [cs, stat].

[175] Karen Simonyan and Andrew Zisserman. Very Deep Convolutional Networks for Large-Scale Image Recognition, April 2015. arXiv:1409.1556 [cs].

[176] Henggang Cui, Vladan Radosavljevic, Fang-Chieh Chou, Tsung-Han Lin, Thi Nguyen, Tzu-Kuo Huang, Jeff Schneider, and Nemanja Djuric. Multimodal Trajectory Predictions for Autonomous Driving using Deep Convolutional Networks. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 2090–2096, Montreal, QC, Canada, May 2019. IEEE.

[177] argoverse. Argoverse 2: Motion Forecasting Competition. https://eval.ai/web/challenges/challenge-page/1719/leaderboard/4098, April 2024. Accessed on 28.024.2024.

[178] Interpret Challenge. Interpret challenge. http://challenge.interaction-dataset.com/leader-board, Aug 2022. Accessed on 23.07.2024.

[179] Thomas N. Kipf and Max Welling. Semi-Supervised Classification with Graph Convolutional Networks, February 2017. arXiv:1609.02907 [cs, stat].

[180] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. Graph Attention Networks, February 2018. arXiv:1710.10903 [cs, stat].

[181] Xiaoyu Mo, Yang Xing, and Chen Lv. Graph and Recurrent Neural Network-based Vehicle Trajectory Prediction For Highway Driving, July 2021. arXiv:2107.03663 [cs].

[182] Sepp Hochreiter and Jürgen Schmidhuber. Long Short-Term Memory. *Neural Computation*, 9(8):1735–1780, November 1997.

[183] Jiachen Li, Hengbo Ma, Zhihao Zhang, Jinning Li, and Masayoshi Tomizuka. Spatio-Temporal Graph Dual-Attention Network for Multi-Agent Prediction and Tracking, February 2021. arXiv:2102.09117 [cs].

[184] Hang Zhao, Jiyang Gao, Tian Lan, Balakrishnan Varadarajan, Yue Shen, Cordelia Schmid, Congcong Li, Chen Sun, Benjamin Sapp, Yi Shen, Yuning Chai, and Dragomir Anguelov. TNT: Target-driveN Trajectory Prediction, 2020.

[185] Daniel Grimm, Philip Schörner, Moritz Dreßler, and J.-Marius Zöllner. Holistic Graph-based Motion Prediction. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2965–2972, London, United Kingdom, May 2023. IEEE.

[186] Zhiqian Lan, Yuxuan Jiang, Yao Mu, Chen Chen, and Shengbo Eben Li. SEPT: Towards Efficient Scene Representation Learning for Motion Prediction, December 2023. arXiv:2309.15289 [cs].

[187] Justin Gilmer, Samuel S. Schoenholz, Patrick F. Riley, Oriol Vinyals, and George E. Dahl. Neural Message Passing for Quantum Chemistry, June 2017. arXiv:1704.01212 [cs].

[188] Aynaz Taheri and Tanya Berger-Wolf. Predictive temporal embedding of dynamic graphs. In *Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*, pages 57–64, Vancouver British Columbia Canada, August 2019. ACM.

[189] João Rico, José Barateiro, and Arlindo Oliveira. Graph Neural Networks for Traffic Forecasting, April 2021.

[190] Lukas Birkemeyer, Julian Fuchs, Alessio Gambi, and Ina Schaefer. SOTIF-Compliant Scenario Generation Using Semi-Concrete Scenarios and Parameter Sampling, August 2023. arXiv:2308.07025 [cs, eess].

[191] Daniel J. Fremont, Tommaso Dreossi, Shromona Ghosh, Xiangyu Yue, Alberto L. Sangiovanni-Vincentelli, and Sanjit A. Seshia. Scenic: A Language for Scenario Specification and Scene Generation. In *Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 63–78, June 2019. arXiv:1809.09310 [cs].

[192] Moritz Klischat and Matthias Althoff. Synthesizing Traffic Scenarios from Formal Specifications for Testing Automated Vehicles. In *2020 IEEE Intelligent Vehicles Symposium (IV)*, pages 2065–2072, Las Vegas, NV, USA, October 2020. IEEE.

[193] Quanyi Li, Zhenghao Peng, Lan Feng, Qihang Zhang, Zhenghai Xue, and Bolei Zhou. MetaDrive: Composing Diverse Driving Scenarios for Generalizable Reinforcement Learning, July 2022. arXiv:2109.12674 [cs].

[194] Lan Feng, Quanyi Li, Zhenghao Peng, Shuhan Tan, and Bolei Zhou. TrafficGen: Learning to Generate Diverse and Realistic Traffic Scenarios, March 2023. arXiv:2210.06609 [cs].

[195] Simon Suo, Sebastian Regalado, Sergio Casas, and Raquel Urtasun. TrafficSim: Learning to Simulate Realistic Multi-Agent Behaviors. In *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10395–10404, Nashville, TN, USA, June 2021. IEEE.

[196] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An Open Urban Driving Simulator, November 2017. arXiv:1711.03938 [cs].

[197] Stephan R. Richter, Zeeshan Hayder, and Vladlen Koltun. Playing for Benchmarks. In *2017 IEEE International Conference on Computer Vision (ICCV)*, pages 2232–2241, Venice, Italy, October 2017. IEEE.

[198] Panpan Cai, Yiyuan Lee, Yuanfu Luo, and David Hsu. SUMMIT: A Simulator for Urban Driving in Massive Mixed Traffic. In *2020 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4023–4029, Paris, France, May 2020. IEEE.

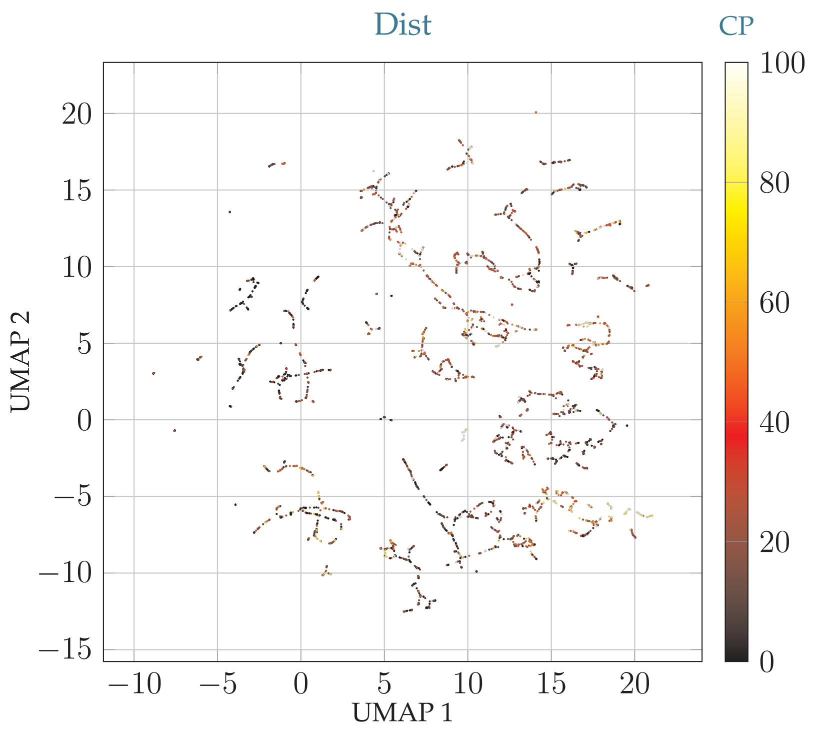
[199] Eugene Vinitsky, Nathan Lichtlé, Xiaomeng Yang, Brandon Amos, and Jakob Foerster. Nocturne: a scalable driving benchmark for bringing multi-agent learning one step closer to the real world, February 2023. arXiv:2206.09889 [cs].

[200] Cole Gulino, Justin Fu, Wenjie Luo, George Tucker, Eli Bronstein, Yiren Lu, Jean Harb, Xinlei Pan, Yan Wang, Xiangyu Chen, John D. Co-Reyes, Rishabh Agarwal, Rebecca Roelofs, Yao Lu, Nico Montali, Paul Mougin, Zoey Yang, Brandyn White, Aleksandra Faust, Rowan McAllister, Dragomir Anguelov, and Benjamin Sapp. Waymax: An Accelerated, Data-Driven Simulator for Large-Scale Autonomous Driving Research, October 2023. arXiv:2310.08710 [cs].

[201] J. P. Verma and Priyam Verma. *Determining Sample Size and Power in Research Studies: A Manual for Researchers*. Springer Singapore, Singapore, 2020.

Appendix

1 Scene Space Analysis for Criticality Metric



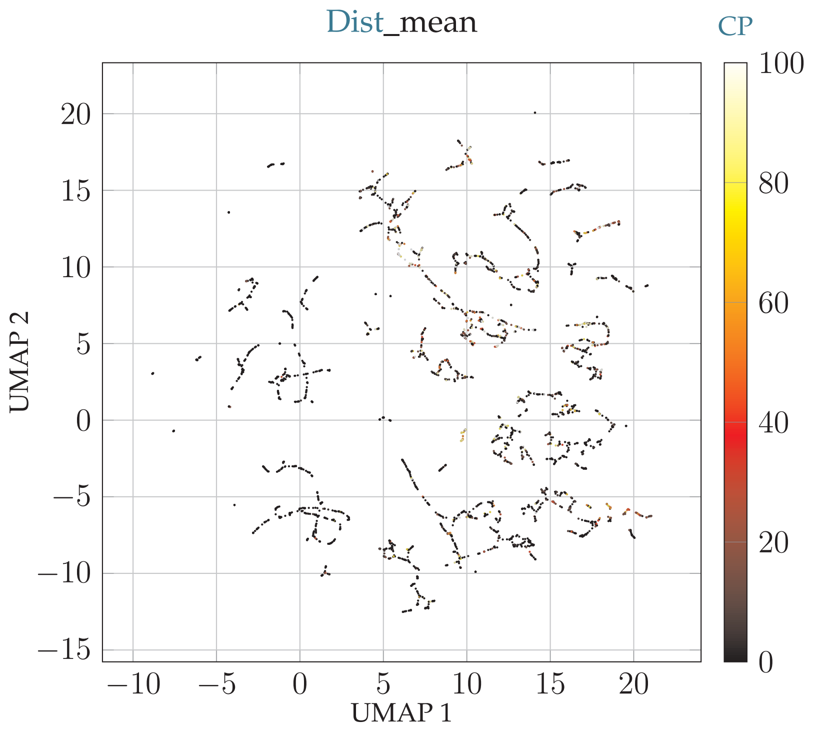
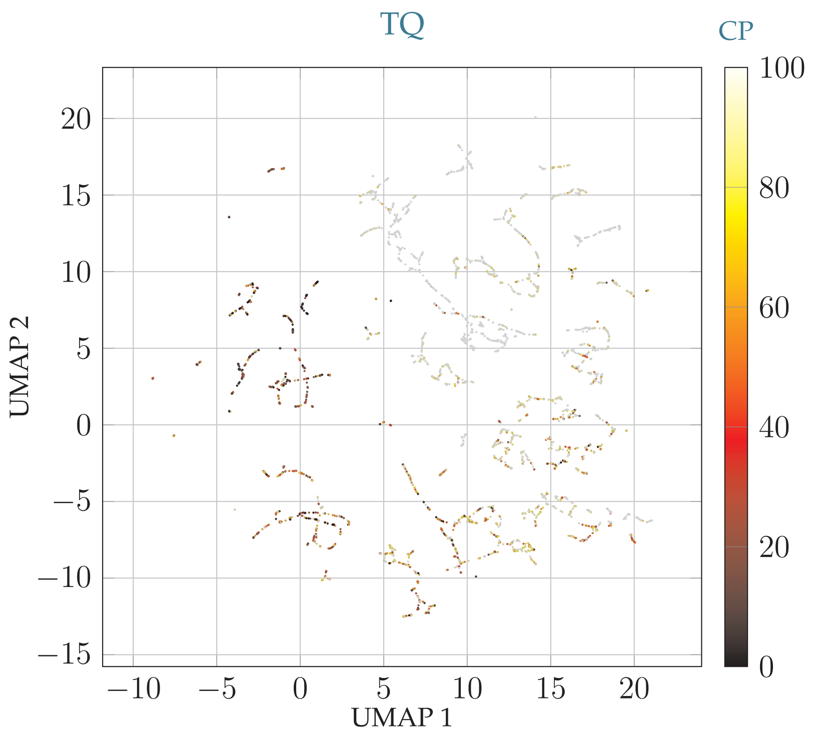


Figure A.1: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for Dist.



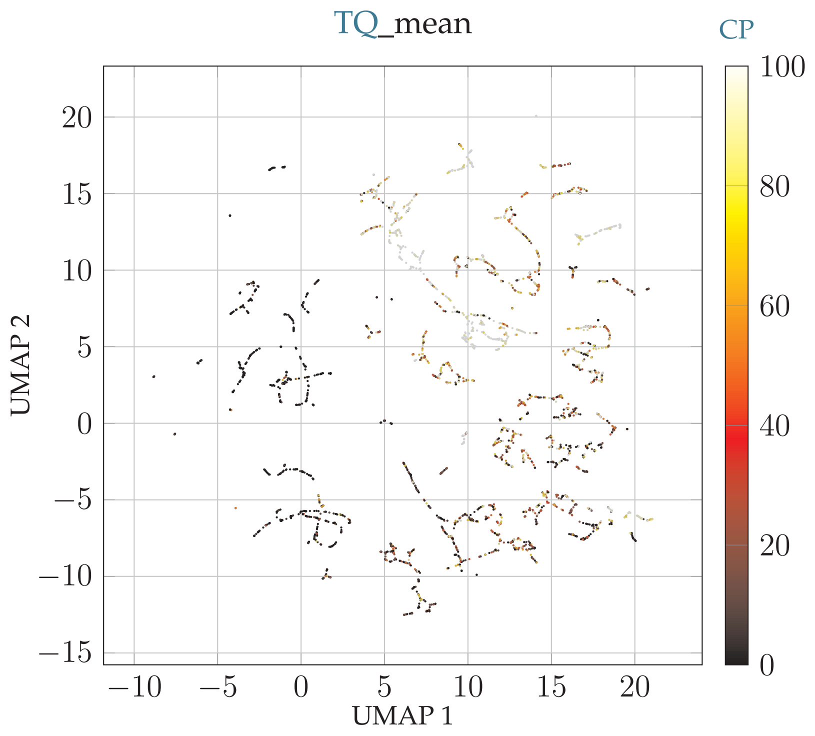
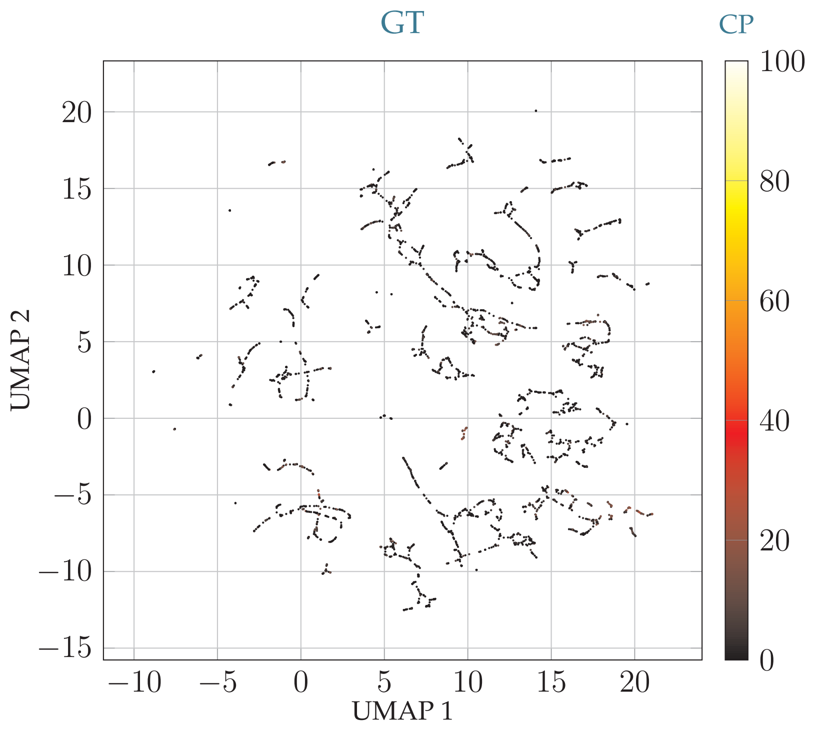


Figure A.2: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for TQ.



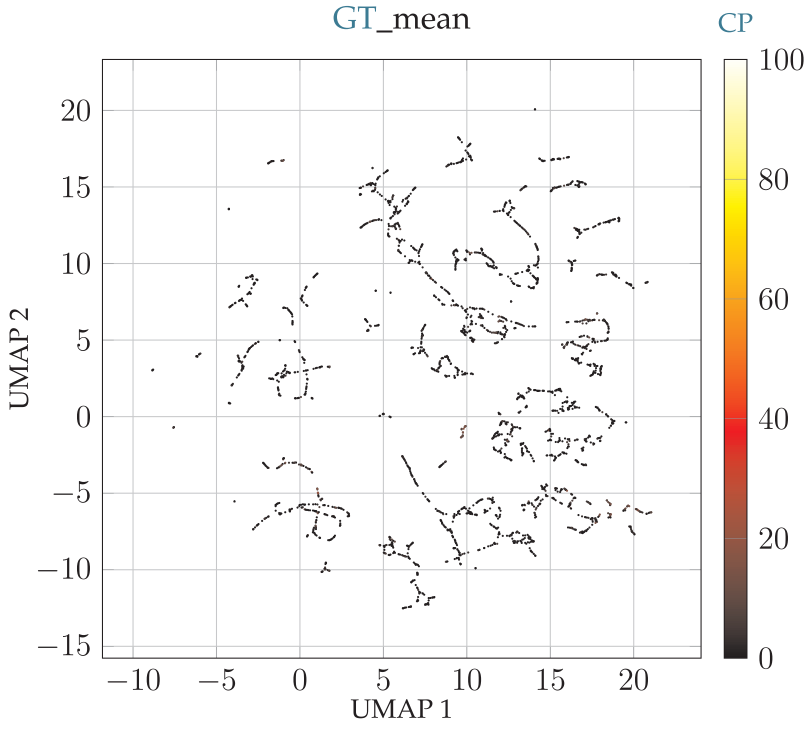
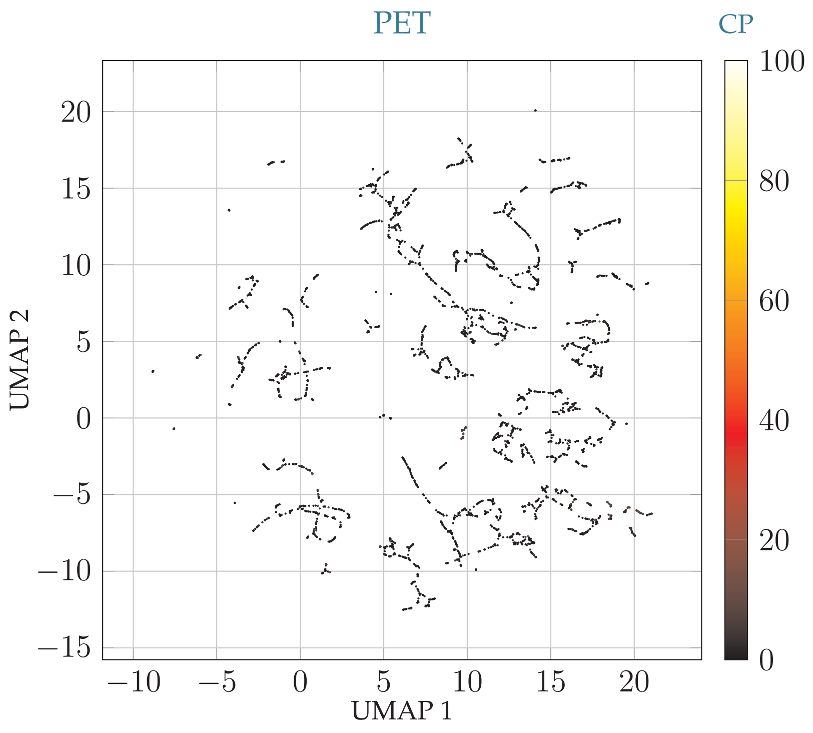


Figure A.3: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for GT.



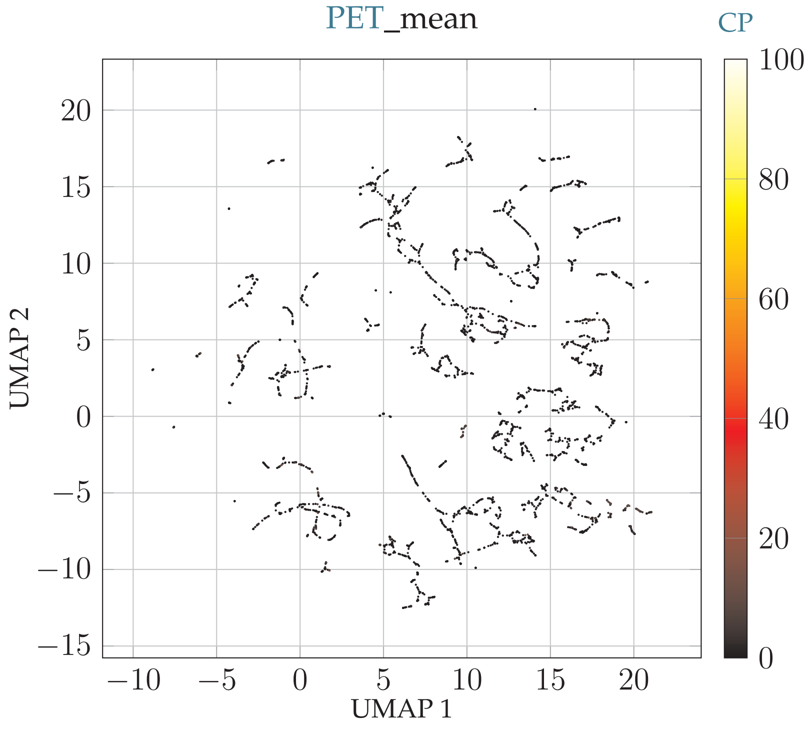
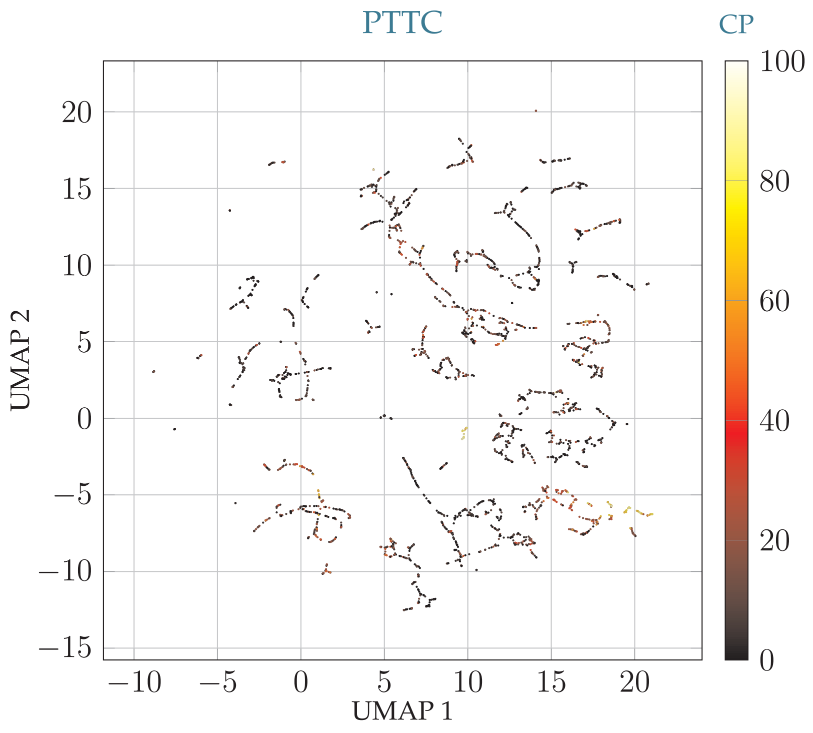


Figure A.4: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for PET.



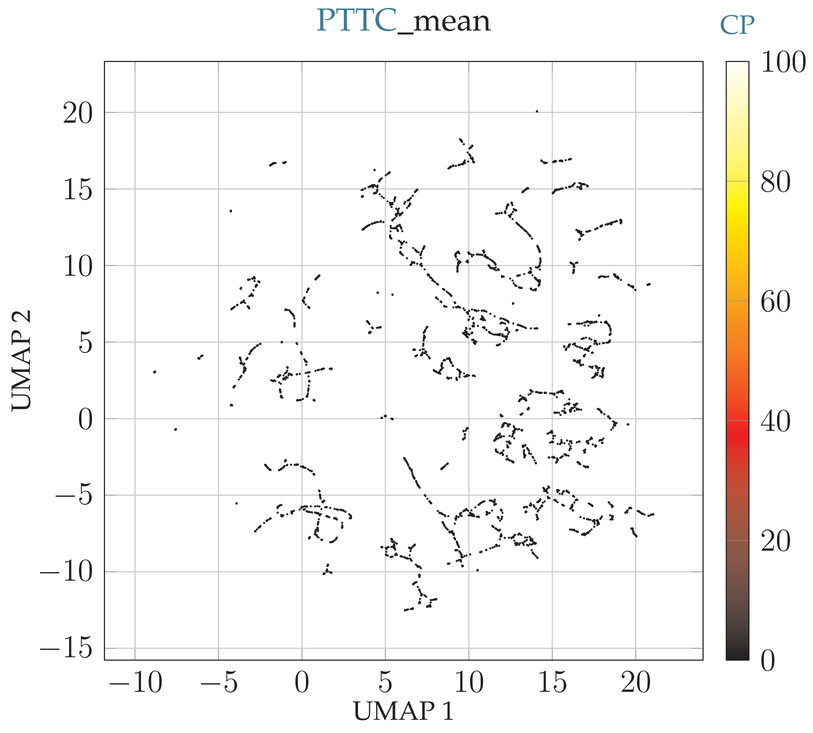
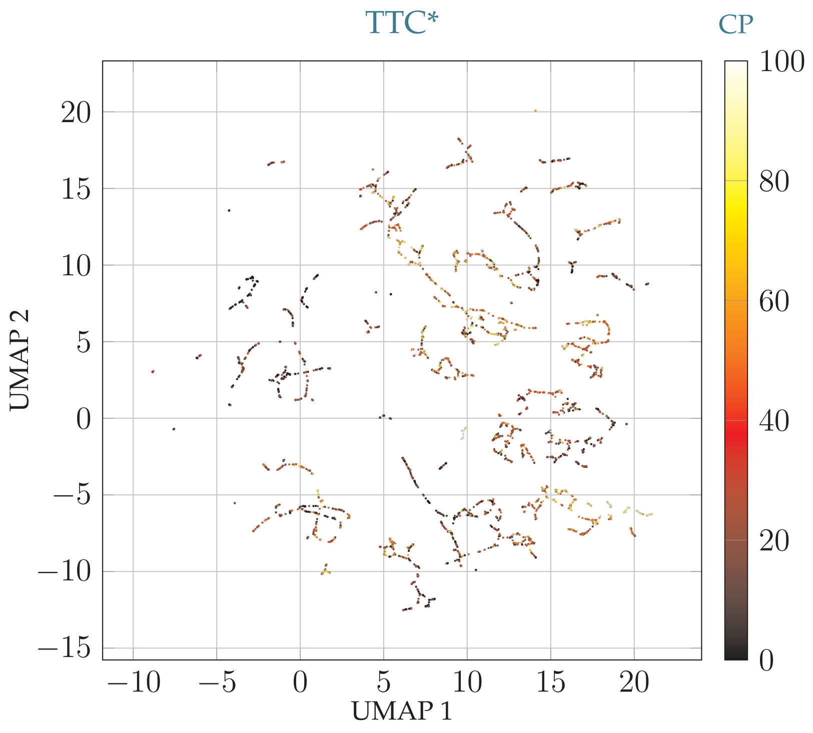


Figure A.5: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for PTTC.



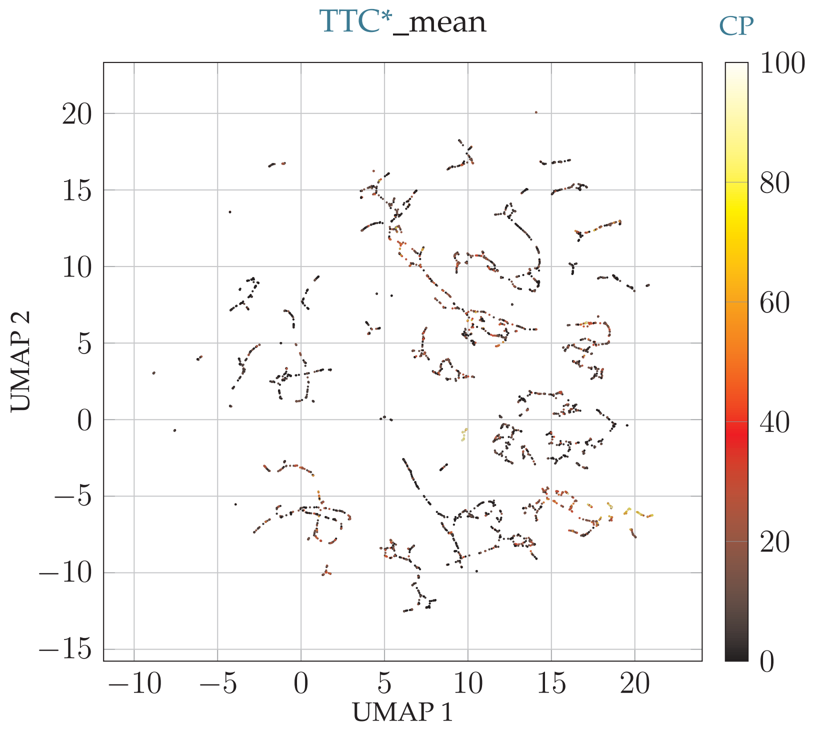
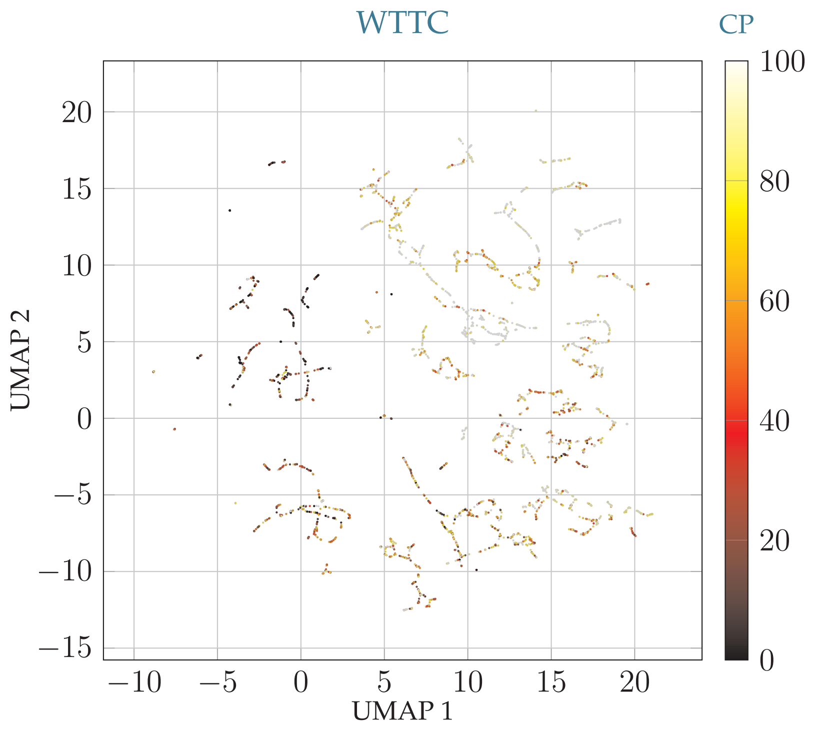


Figure A.6: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for TTC.



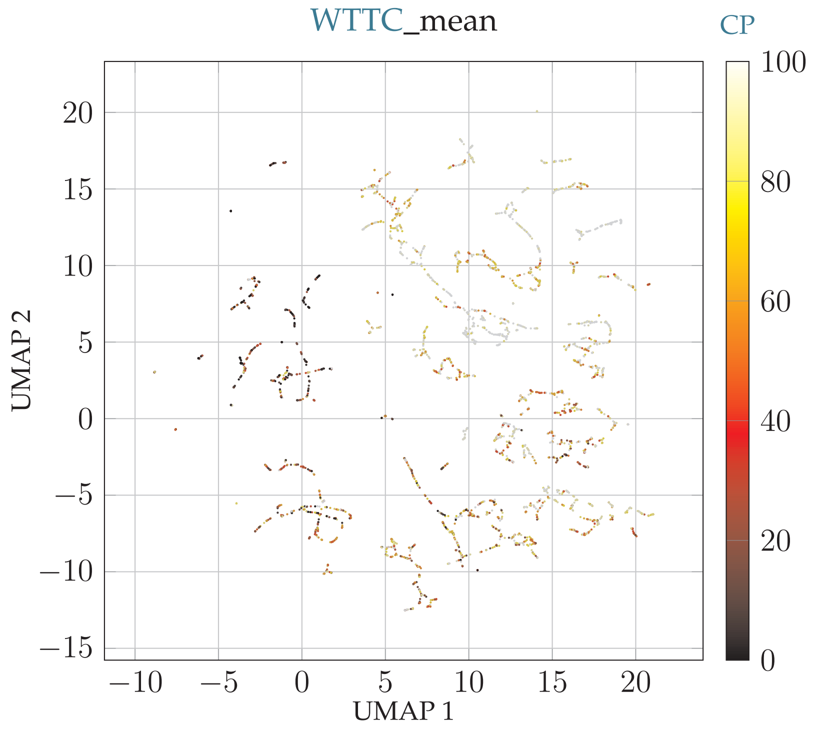


Figure A.7: 2D-reduced scene embedding space (cf. Section 7.4): A seed-scene represents a data point, with the color (0 ... 100) of the point indicating the CP for WTTC.

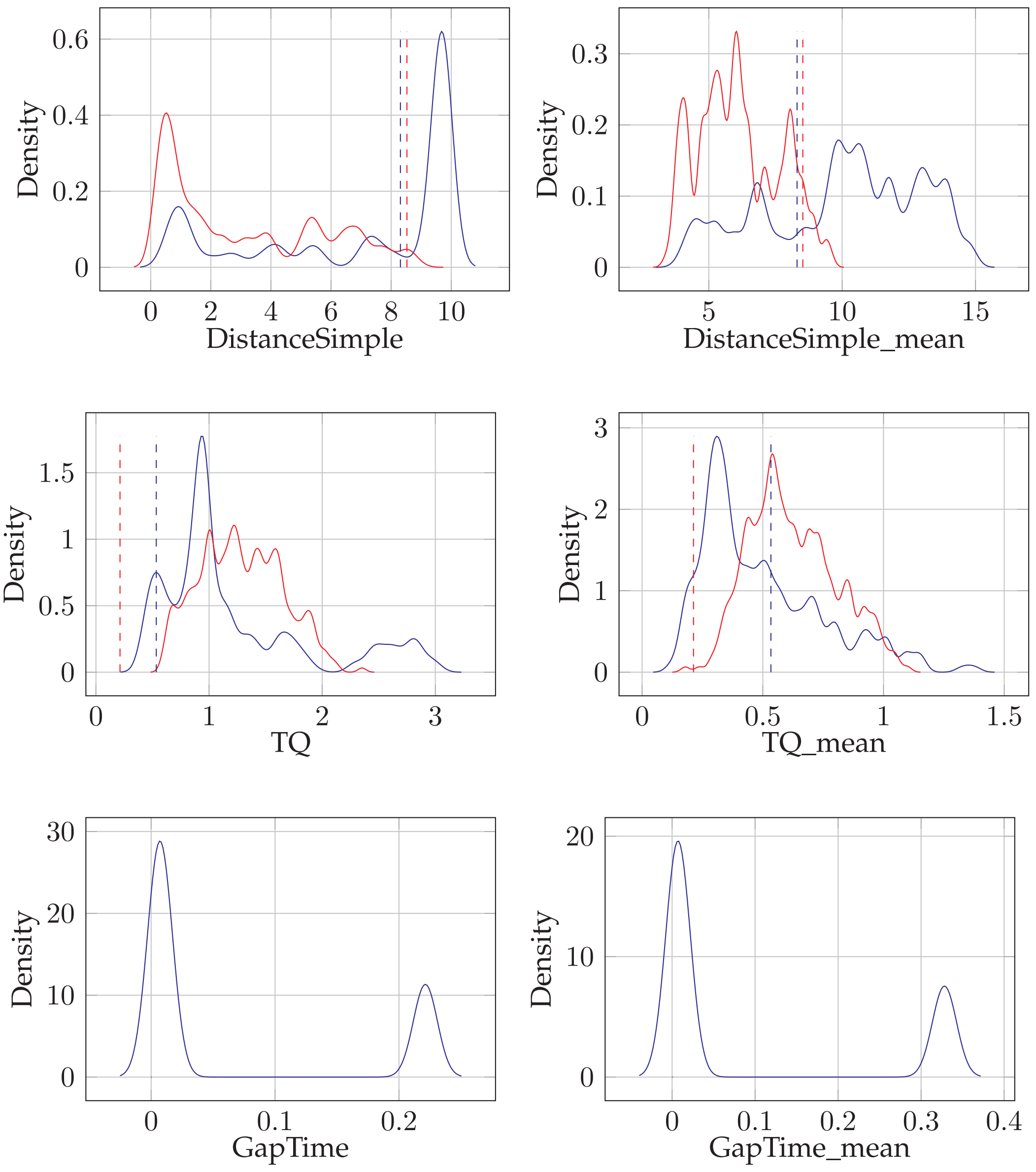


Figure A.8: The blue and red colors indicate the distribution of criticality for all child-simulations of two seed-scenes depicted in Figure 7.4. The dashed lines marking the measured worst cases in the real situation respectively.

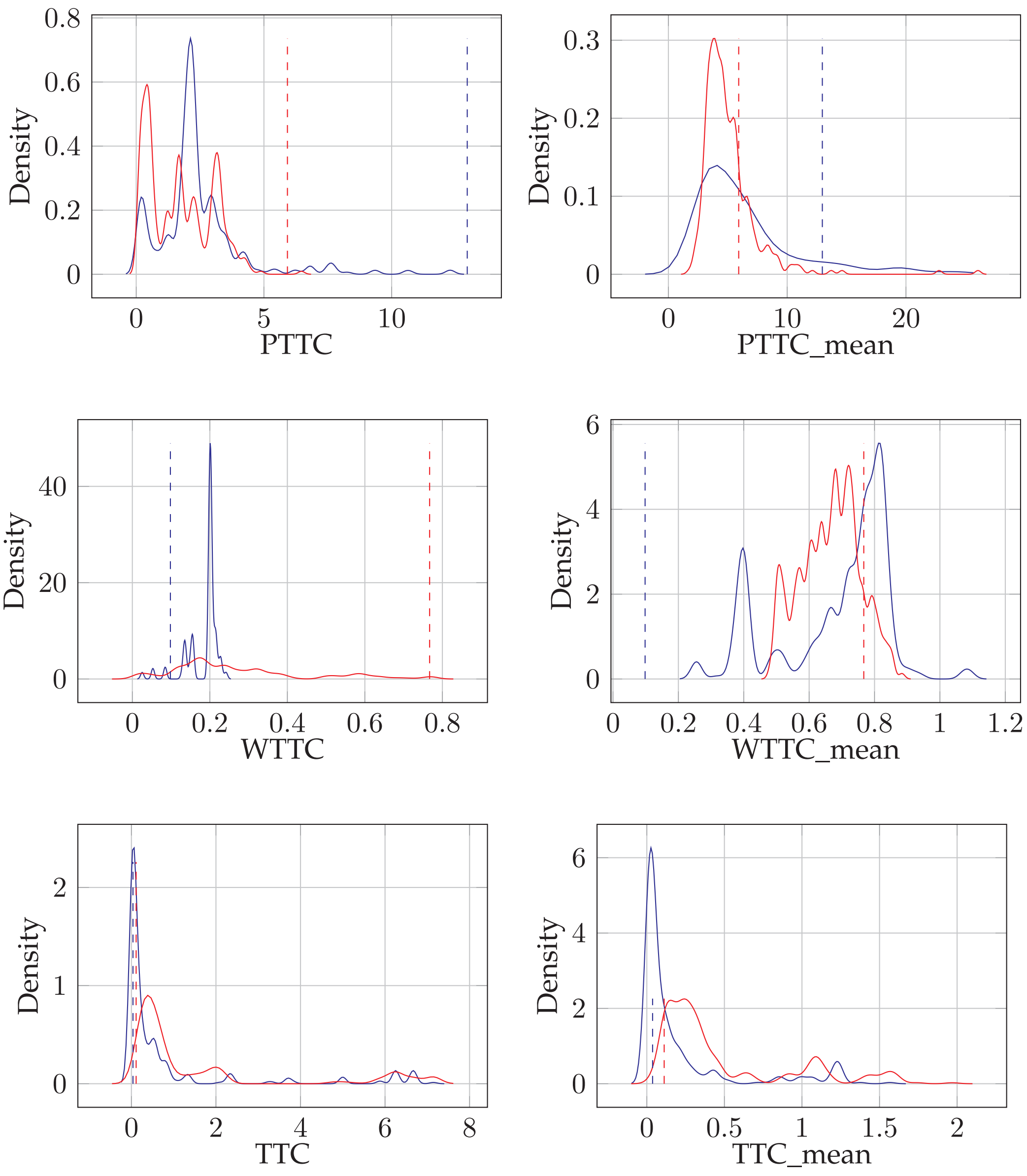


Figure A.9: The blue and red colors indicate the distribution of criticality for all child-simulations of two seed-scenes depicted in Figure 7.4. The dashed lines marking the measured worst cases in the real situation respectively.

3 Criticality Potential for Clusters

Cluster	CP Dist	CP GT	CP PET	CP PTTC	CP TQ	CP TTC	CP WTTC	overall CP	variance
15	69.45	10	3.79	46.88	93.38	70.42	91.13	41.96	244
28	53.57	0.99	0.32	13.76	99.39	44.08	90.81	38.13	216
0	43.97	3.12	0.72	12.31	97.47	47.44	94.33	37.51	301
6	43.83	1.31	0.25	13.66	98.58	47.87	88.12	36.5	239
30	60.44	12.66	5.66	48.23	80.84	68.45	69.14	35.4	521
10	39.15	1.96	0.49	8.7	97.59	38.34	82.35	32.94	327
11	40.83	2.38	0.73	13.79	93.62	43.41	88.29	31.98	274
5	36.33	1.87	0.26	6.81	95.09	34.57	93.01	31.87	247
16	35.28	1.23	0.29	8.1	94.41	43.4	81.64	30.57	311
14	36.51	2.19	0.51	6.55	88.23	30.07	85.15	27.86	371
3	44.78	4.19	1.55	24.65	81.28	51.03	77.13	27.48	333
8	33.43	1.36	0.44	10.19	90.44	36.62	77.05	27.08	311
18	26.42	1.36	0.35	7.99	90.71	29.58	90.72	26.76	146
21	24.65	0.74	0.14	6.43	91.03	28.06	78.52	24.9	267
23	34.7	1.97	0.19	5.79	82.78	23.24	82.89	24.63	340
24	31.71	1.99	1.15	7.11	78.44	30.74	80.37	23.93	415
2	23.59	1.73	0.51	5.68	76.17	23.6	70.18	20.79	401
1	32.29	3.08	1.35	15.52	67.31	34.62	63.54	20.2	408
7	29.16	1.54	0.68	8.56	74.41	27.39	66.29	20.14	361
19	34.54	5.23	2.61	24.42	54.53	41.62	48.98	19.59	502
12	18.03	1.15	0.16	3.63	69.34	15.83	63.61	16.02	348
13	27.36	4.66	2.22	17.77	43.1	32.35	44.56	15.12	375
9	19.33	1.04	0.3	4.45	56.73	15.99	52.91	15.02	509
27	19.44	0.91	0.16	4.86	53.22	13.8	54.39	13.73	404
20	19	2.45	0.78	9.3	43.64	20.89	51.58	13.22	451
29	9.92	0.65	0.08	3.39	49.4	8.98	53.55	12.39	417
31	11.19	2.4	0.65	5.78	34.16	13.17	32.93	9.2	339
22	15.53	2.22	1.14	8.64	23.81	19.73	22.46	8.01	187
17	15.55	2.65	1.28	8.5	24.18	16.63	15.72	7.62	218
26	6.61	0.65	0.45	2.03	26.54	5.33	28.63	6.23	284
25	10.42	2.66	1.54	5.37	16.84	12.37	21.5	6.19	199
4	8.5	0.78	0.2	3.38	20.7	9.07	19.39	5.38	224

Table A.1: Evaluation of the different clusters from the test space reduced to 2D in Figure 7.6. The CP values of each scene for each metric are averaged over the entire cluster. The clusters are ordered according to the overall CP

4 Simulation Time Analysis

This section provides a brief overview of the simulation times with the individual behavior models. Figure A.10 shows the various modules and behavior models used in the simulation. The preparation of the seed-scene includes all loading cycles of trajectory histories and preparation steps, such as the deterministic permutation of the behavior models to simulate a seed scene. However, this step only needs to be carried out once for each seed scene. The preparation and re-initialization of all driver models and the preparation of the logger for each child-scenario (Prep. of child-scenario) is repeated. The driver models are sometimes executed several times within a simulation step and several runs in a child-scenario as a result. Particularly noteworthy is the hybrid model, which requires an average of 150 ms for each prediction of a trajectory. The graph model (Relation-based model) creates a joint prediction, whereby all vehicles in the scene are predicted, which means that the number of vehicles actually to be predicted is insignificant. Assuming an average of 11 traffic participants in a scene, the simulator requires an average of around 389 ms for a prediction step if all models are assigned equally. Depending on which models were selected for a child scenario, this can take from 50 ms for simple models to up to 9 s for exclusively hybrid models. It can be clearly seen here that the inference of the ML models forms a clear bottleneck in the simulation chain.

The tests were carried out on a system with an Intel(R) Core(TM) i7-7800X CPU @ 3.50GHz and a GeForce GTX 1080 Ti GPU

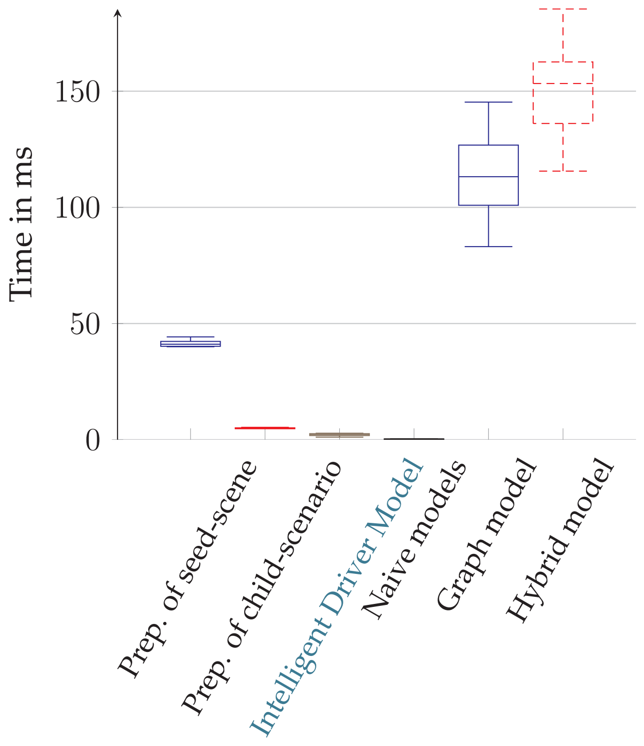


Figure A.10: Execution time analysis of different modules of the simulation

5 Fine-Tuning Driver Behaviors on Critical Scenarios

The criticality metrics applied to the scene space in Chapter 5 and Chapter 7 lead to an evaluation of traffic scenes based on their criticality. These findings can be used to selectively train the imitation learning models presented in Chapter 6. This chapter examines the effects of a behavior model trained on specifically critical traffic scenarios on the evaluation seed-scenes.

The Hybrid-model is employed in the training process. The model is pre-trained in accordance with the methodology outlined in Section 6.2.1, utilizing a large training dataset comprising a heterogeneous set of random scenarios. The training process is terminated after 15 epochs. In the second step, the model is fine-tuned using only critical scenarios. The most critical 150 scenes, as indicated by their average CP value, were identified from the scene space depicted in Figure 7.6. Due to the inherent limitations of CP as a metric for assessing the criticality of individual traffic participants and the potential for criticality to shift within a given scenario based on the behavior model, all traffic participants present in the selected critical scenes were utilized as input for the fine-tuning process. This resulted in a total of 1438 training samples. In addition, a reference model is retrained on random, not exclusively critical scenarios.

In the next step, the methodology from Section 7.2 is applied with the model fine-tuned to the critical scenes and the reference model. Only the *Hybrid1* model from Table 7.1 is replaced by the fine-tuned models, all other behavior models remain and are utilized in the simulation of the child-scenarios.

In the evaluation of the simulation results, it is ensured that the seed-scenes that were extrapolated were not included in the training dataset for fine-tuning. Figure A.11 depicts a seed-scene with three exemplary metrics. The blue curve represents the criticality distribution of a seed scene in which one of the models is explicitly fine-tuned to critical scenes. The red curve illustrates the criticality distribution of the simulated child-scenarios of the same seed-scene with the reference model. It can be seen that the frequency of critical child-scenarios is generally higher with the fine-tuned model compared to the reference model (Dist: shift to the left, TQ: shift to the right, TTC*: shift to the right). This tendency can also be seen in the CP changes in Appendix 5. The CP values for intersection-specific metrics such as GT and PET have not increased at all or only slightly. However, a positive trend can be seen for all other metrics. For example, the CP value of the Dist metric increased by 3.81. The collisions in the child scenarios were also examined as part of the analysis. For the reference model, 5633 collisions were identified, while the

Metric name	CP change
DistanceSimple	+3.81
DistanceSimple_mean	+3.09
GapTime	+0.34
GapTime_mean	+0.17
PET	+0.01
PET_mean	+0.0
PTTC	+2.71
PTTC_mean	+1.64
TQ	+1.64
TQ_mean	+0.99
WTTC	+2.2
WTTC_mean	+1.84
TTC	+3.68
TTC_mean	+2.97

Table A.2: Average CP change over the test dataset

simulation run with the refined model yielded 5791 collisions. Accordingly, the collision frequency is observed to increase by 2.8%.

The model fine-tuned to critical scenarios generates more critical scenarios. This can be due to the fact that situations with higher criticality were very strongly represented in the training samples. The distances were decreased, and relative speeds were presumably increased compared to the average behavior. As a result, this model has learned to behave in such a way as to take less of a safe distance. This analysis demonstrates that a single behavior has a large impact on the resulting scenario. However, it also shows that the composition of the behavior models used in the simulation has a strong influence on the criticality distribution.

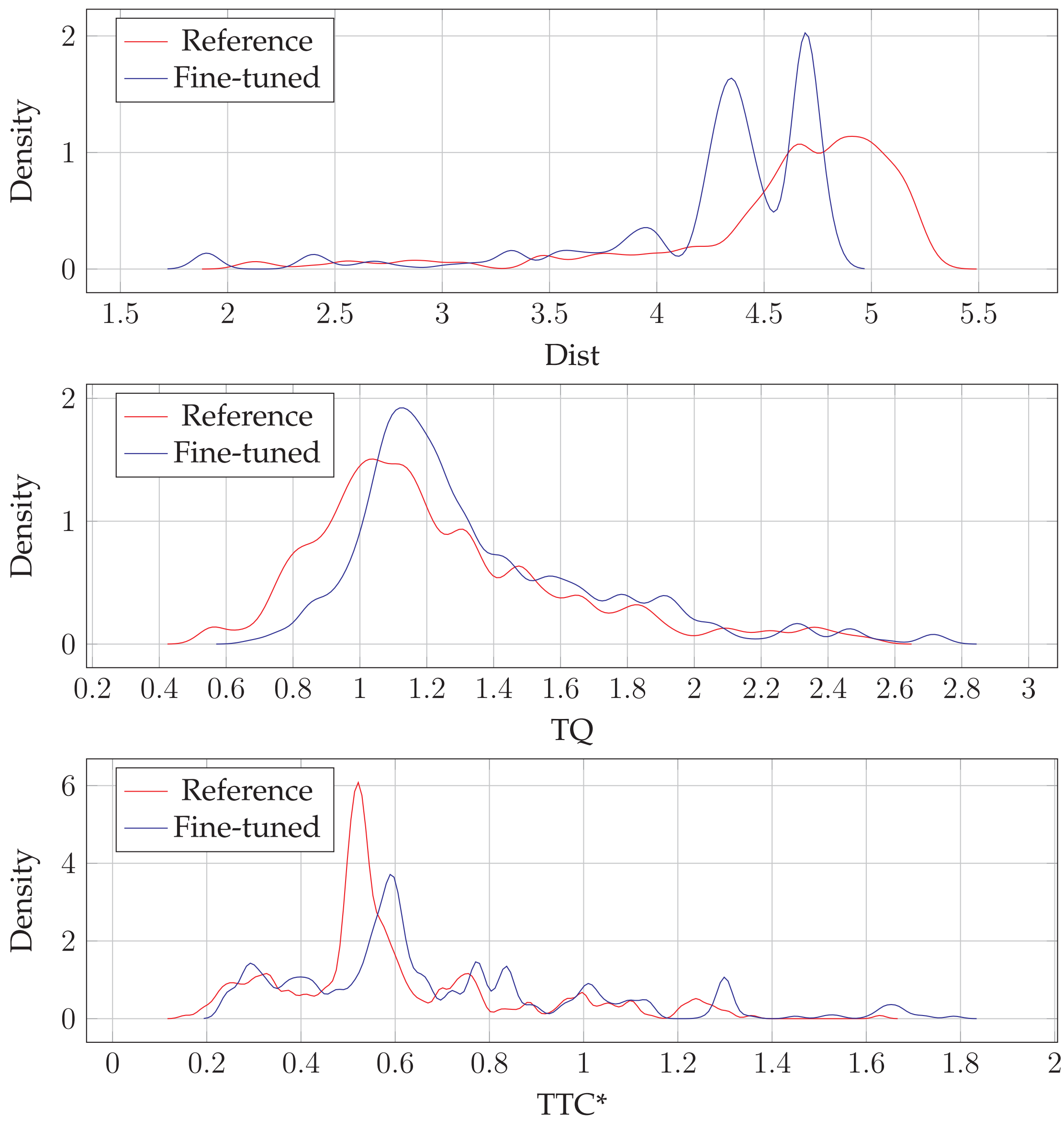


Figure A.11: Comparison of the criticality distribution of three exemplary metrics (Dist, TQ, TTC*) of one seed-scene between the fine-tuned model (blue) and the reference (red)