Carl von Ossietzky Universität Oldenburg
Fakultät II - Informatik, Wirtschafts- und Rechtswissenschaften
Department für Informatik

From Real-World Traffic Data to Scenarios in the Context of Automated Vehicles

Von der Fakultät für Informatik, Wirtschafts- und Rechtswissenschaften
der Carl von Ossietzky Universität Oldenburg zur Erlangung des Grades und Titels

Doktor der Ingenieurswissenschaften (Dr.-Ing.)

angenommene Dissertation

von **Herrn Lars Klitzke**
geboren am 05.05.1989 in Nordhorn

Gutachter:
Prof. Dr. Frank Köster
Prof. Dr.-Ing. habil. Jorge Marx Gómez

Tag der Disputation:
02. Februar 2026

Danksagung

Dass ich diesen akademischen Weg gehen konnte und an solch spannenden Themen arbeiten darf, verdanke ich so vielen Wegbegleitern. Ihr habt auf so viele verschiedene Arten dazu beigetragen, dass diese Arbeit entstanden ist. Dafür möchte ich mich von Herzen bei euch bedanken.

Ganz besonders bedanken möchte ich mich bei Prof. Dr. Carsten Koch und meinem Doktorvater Prof. Dr. Frank Köster, da es diese Arbeit ohne Euch nicht geben würde. Ich danke Dir, Carsten, für die Unterstützung und Förderung während meiner Zeit als Wissenschaftlicher Mitarbeiter an der Hochschule Emden/Leer und auch für die Zeit darüber hinaus. Frank, danke, dass Du dir stets Zeit zum Austausch genommen hast und für die vielen Hilfestellungen auf sowohl fachlicher, als auch menschlicher Ebene. Ich schätze den Austausch mit Euch beiden sehr.

Ebenfalls bedanke ich mich bei meinem Zweitgutachter Prof. Dr.-Ing. habil. Jorge Marx Gómez für den guten und angenehmen Austausch, sowie bei Prof. Dr.-Ing. Jürgen Sauer und Dr. André Bolles für die Mitwirkung in der Prüfungskommission.

Der Wechsel von der Hochschule Emden/Leer zum Deutschen Zentrum für Luft- und Raumfahrt (DLR) hat maßgeblich diese Arbeit geprägt, sowohl was die Tiefe der Dissertation als auch deren Anwendbarkeit an reale Probleme betrifft. Mein Dank gilt meinen DLR-Kolleg:innen am Institut für Verkehrssystemtechnik: Die fachübergreifenden Diskussionen und Impulse, als auch die Möglichkeit, meine Ansätze direkt in Projekten zu erproben, haben sehr dazu beigetragen, die entwickelten Konzepte zu schärfen und zu validieren. Hierbei danke ich insbesondere meiner Forschungsgruppe „Modellierung des Verkehrsgeschehens“.

An dieser Stelle möchte ich auch meiner Familie und Freunden danken. Ganz besonders für das Mitfiebern und den Rückhalt von Soffi, Dennis, Lina, Tobi, Ina, Daniel, Kirsten und Kevin: Danke und fühlt euch gedrückt! Außerdem möchte ich mich ganz herzlich bei meinen Eltern bedanken. Danke, dass ihr mich ermutigt habt, diesen Weg einzuschlagen, und für eure immerwährende Unterstützung. In vielerlei Hinsicht ist diese Arbeit auch euer Verdienst.

Abschließend möchte ich mich von Herzen bei meiner Frau Svenja und bei unserem flauschigen Vierbeiner Eddi bedanken. Svenja, ich habe es bereits bei unserer Hochzeit gesagt und kann es hier nur bekräftigen: Ich bin sehr dankbar, dich an meiner Seite zu haben. Ich bin stolz darauf, wie wir in herausfordernden Zeiten zusammenhalten und aufeinander vertrauen können, ebenso wie darauf, die guten Zeiten gemeinsam zu genießen. Ich danke dir von Herzen für dein Verständnis und deine Liebe und freue mich auf unseren nächsten gemeinsamen Lebensabschnitt.

Abstract

The large-scale introduction of automated vehicles (AVs) on public roads is an ambitious and challenging goal. This technology aims to significantly contribute to increased traffic safety and comfort, while also serving as the foundation for further innovative mobility concepts. But, one of the biggest challenges in introducing such systems is ensuring their compliance with regulatory requirements, since it is essential that these systems function correctly and safe. This is particularly demanding for higher-level automated vehicles, which must navigate public roads safely without a human fallback option.

Within the framework of homologation of automated vehicles, an approach has been established where the driving function or the automated vehicle is tested in specific *scenarios*. However, the availability of an extensive dataset with diverse scenarios is a critical prerequisite. Theses scenarios can be defined based on different data sources. One method for collecting and evaluating scenarios involves using real traffic data. Such data is highly valuable as it realistically reflects the behavior of human road users and also includes atypical behavior or even critical conflicts between participants.

For collecting real traffic data various methods are available, each with different strengths and weaknesses. One method is infrastructure-based traffic data collection using road side units, which main advantage is capturing traffic events comprehensively and over an extended period. This allows for the continuous and simultaneous capture of multiple road users and their interactions. Consequently, this method enables a detailed description of scenarios and the identification of rare phenomena, which are particularly important for validating driving functions. However, this continuous stream of traffic data must be systematically processed to create a comprehensive collection of scenarios.

This thesis presents a methodology for representing traffic data collected in the real world based on scenarios and their systematic identification from real-world traffic data. A hierarchical data model is used for this purpose, which semantically describes traffic data at four different levels of abstraction. Various approaches to defining and identifying phenomena at those abstraction levels using real traffic data are presented. Furthermore, a modular platform is introduced that integrates these different methods to continuously identify and analyze scenarios in real traffic data and various environments.

The procedures and methods presented in this work are individually evaluated using real-world problems. Their integration through the modular platform demonstrates the suitability of the proposed approaches for identifying and analyzing scenarios in different environments and for various research questions. Overall, the results show that the proposed methodology enable the systematic identification and representation of scenarios from real-world traffic data contributing to building a large-scale knowledge base of scenarios.

Zusammenfassung

Die Einführung von automatisierten Fahrzeugen (AVs) auf öffentlichen Straßen ist ein anspruchsvolles und herausforderndes Ziel. Die Technologie soll einen wesentlichen Beitrag dazu leisten, die Verkehrssicherheit zu verbessern und den Fahrtkomfort zu erhöhen. Auch soll sie als Grundlage für innovative Mobilitätskonzepte dienen. Allerdings ist eine der größten Herausforderungen für die Einführung von AVs, sicherzustellen, dass diese konform mit den lokalen regulatorischen Richtlinien sind, da gewährleistet werden muss, dass die Systeme in den vorgegebenen Umgebungen sicher und zuverlässig funktionieren. Dies ist insbesondere bei höher automatisierten Fahrzeugen der Fall, da diese auf öffentlichen Straßen sicher operieren müssen — auch ohne eine menschliche Rückfallebene.

Im Rahmen der Homologation von automatisierten Fahrzeugen hat sich ein Ansatz etabliert, bei dem die Fahrfunktion oder das automatisierte Fahrzeug in bestimmten Szenarien getestet wird. Jedoch ist hierfür ein umfangreicher Datensatz mit diversen Szenarien essenziell. Diese Szenarien können aus verschiedensten Datenquellen stammen. Eine Möglichkeit zur Erhebung und Auswertung von Szenarien besteht in der Verwendung realer Verkehrsdaten. Ein großer Vorteil dieser Variante ist, dass das Verhalten von menschlichen Verkehrsteilnehmenden realitätsnah abgebildet wird, somit auch atypisches Verhalten, das ggf. zu kritischen Konflikten führen kann.

Für die Erhebung von realen Verkehrsdaten stehen verschiedene Methoden zur Verfügung, wie bspw. infrastrukturell mittels sogenannter *Roadside Units*. Der Vorteil dieser Methode ist es, dass das Verkehrsgeschehen umfassend, langfristig und kontinuierlich erfasst werden kann. Sie erlaubt somit eine detaillierte Beschreibung von Szenarien sowie die Identifikation seltener Phänomene. Allerdings muss dieser Datenstrom systematisch prozessiert werden, um einen umfangreichen Katalog von Szenarien aufzubauen.

Die vorliegende Arbeit präsentiert eine Methodik zur Darstellung von Szenarien und deren systematischen Extraktion aus real erhobenen Verkehrsdaten. Hierfür wird ein hierarchisches Datenmodell eingesetzt, welches Verkehrsdaten auf vier verschiedenen Abstraktionsebenen semantisch beschreibt. Es werden unterschiedliche Ansätze zur Definition und Identifikation von Phänomenen auf diesen Ebenen vorgeschlagen. Zudem wird eine modulare Plattform vorgestellt, die diese Methoden integriert, um kontinuierlich Szenarien in realen Verkehrsdaten unter verschiedenen Umgebungen zu identifizieren.

Die vorgestellten Verfahren und Methoden werden jeweils separat anhand realer Probleme evaluiert. Ihre Integration durch die modulare Plattform zeigt die Eignung der Ansätze zur Identifikation und Analyse von Szenarien in unterschiedlichen Umgebungen und für verschiedene Fragestellungen. Insgesamt zeigen die Ergebnisse, dass die vorgeschlagenen Methodiken es erlauben, infrastrukturell erhobene Verkehrsdaten systematisch in Szenarien zu überführen und somit einen Beitrag für den Aufbau einer umfassenden Wissensbasis von Szenarien aus realen Verkehrsdaten liefert.

Contents

List of Figures

List of Tables

List of Acronyms

ABox	assertional box 102, 107, 110, 111, 113, 115, 116, 119, 126
ADF	automated driving function 151
ADTF	Automotive Data and Time-triggered Framework 150, 151
AIM	Application Platform for Intelligent Mobility xiii–xv, xxix, 14, 16, 25, 26, 31, 37, 62, 68, 69, 97, 101, 115, 117, 119, 126, 149, 150, 152, 153, 155, 158, 168, 169, 171
API	application programming interface xxiv, 128, 138, 150, 152, 153, 165
ASAM	Association for Standardization of Automation and Measuring Systems 151
AV	automated vehicle vii, 1–5, 7, 20, 21, 24, 25, 124, 170
CDM	Confusion Difference Matrix 93, 95
CI/CD	continuous integration and deployment 136
CSV	comma-separated values 143, 145, 150
DAG	Directed Acyclic Graph xvi, 64, 67, 68, 71, 72, 74, 75, 78, 80, 107, 115–117, 119
DAGMaR	Directed Acyclic Graph of Mapped Roads xiv, 61, 64, 66–68, 71, 72, 74–76, 78, 149
DBMS	database management system 141
DL	Description Logic 102
DLR	German Aerospace Center xiii–xv, xviii, xxix, 14, 16, 26, 31, 37, 62, 68, 69, 97, 101, 115, 117, 149, 152, 153, 160, 165, 168, 169, 171
DTW	Dynamic Time Warping xiv, 29, 35, 51–53, 79
ETL	extract-transform-load 131
EV	electric vehicle 1
FOT	Field Operations Test 8
GMM	Gaussian Mixture Model 35, 43, 81, 88, 94, 96
GNSS	Global Navigation Satellite System 15
GPS	Global Positioning System 15, 150, 151
GT	Gap Time 27, 28
HAC	Hierarchical Agglomerative Clustering 81, 88, 94
HMM	Hidden Markov Model xiv, 29, 34, 35, 43–47, 56, 79, 161, 169

HTTP	Hypertext Transfer Protocol 138
HTTPS	Hypertext Transfer Protocol Secure 138
KB	Knowledge Base 29, 101–104, 110
MOM	message-oriented middleware xvii, 132, 135, 136
NDD	Naturalistic Driving Data 5
NDS	Naturalistic Driving Study 8
OD	Operational Domain 170
ODD	Operational Design Domain 2, 4, 24
ORM	Object-relational Mapping xvii, 127, 138–141, 145
OWL	Web Ontology Language 102, 103, 107, 110
PET	Post Encroachment Time xviii, 28, 117, 125, 126, 130, 146, 148, 155–157, 162–164
RDB	relational database 103, 127, 131, 132, 138–142, 145
RDF	Resource Description Framework xvi, 102–104
REST	Representational State Transfer 138, 152
ROS	Robot Operating System 151
SBT	scenario-based testing 3, 4, 19, 23, 25, 27, 29
Scenimini	Scenario mining platform 127–129, 135, 137, 138, 148, 153, 154, 157–162, 168–170
ScenORM	Scenario object-relational mapping xviii, 127, 138–142, 145, 155–157, 162, 165
SMoS	Surrogate Measure of Safety 27–29, 117, 126, 146, 148, 155, 157, 164, 171
SMP	Scenario mining pipeline xvii, xviii, 127–142, 148, 150, 152, 154–161, 165, 168, 170
SOA	service-oriented architecture xvii, 130–133, 137
SPARQL	SPARQL Protocol And RDF Query Language 29, 103, 104, 107, 110, 111, 126
SQL	Structured Query Language 138
SSCS	space-sharing conflict scenario 11, 24, 25, 101, 104–107, 109–113, 115, 117–119, 126, 169
SWRL	Semantic Web Rule Language 110, 113
TAP-API	Traffic Processing application programming interface (API) 128, 137–139, 152, 158, 161, 162
TASI	Traffic situation analysis and interpretation xvii, xviii, 127, 138, 142, 143, 145–152, 165, 168, 171
TBox	terminology box 102, 107, 108, 111, 112, 116, 119, 126
TraSceAPI	traffic and scenario API 138, 152, 158, 159, 161
TTC	time to collision 27, 28, 148
URI	Unique Resource Identifier 102, 104
UTM	Universal Transverse Mercator xiii, xxvii, 15–17, 62, 142, 143
VOWL	Visual Notation for OWL Ontologies 103, 107
VuT	Vehicle under Test 2, 8, 9, 31, 127, 150

VVM	Validation and Verification Methods 153, 154, 165
WFS	Web Feature Service 127
WGS	World Geodetic System 15
WMS	Web Map Service 127
XML	eXtensible Markup Language 113–116
XSL	eXtensible Stylesheet Language xxix, 113–115

List of Symbols

Numbers and variables

a The acceleration of a road user typically expressed in $\frac{m}{s^2}$ 13
y A component of the (partial) scene in the driving primitive domain(s). 36, 47–49, 52, 163
T The number of elements of a time-series. 14, 36, 45, 52
u The easting component of a position in the UTM coordinate system. 16
N The number of elements of a set in the local context. 35, 36, 61, 67, 68
m A component of the maneuver signature definition. 36
μ The mean of a sequence or tuple $s = s_1, s_2, \ldots, s_N$ in a specific context defined as $\mu = \frac{1}{N} \sum_i^{|s|} s_i$ 46, 47
v The northing component of a position in the UTM coordinate system. 16
ω The orientation of a road user expressed in radians 13, 62, 63
r A road. 61, 63, 64, 67, 68, 88
t The time. 13, 14, 16, 17, 27, 28, 34, 36, 61–68, 163
v The velocity of a road user typically expressed in $\frac{m}{s}$ 13, 27

Vectors

$\mathbf{y}$ The (partial) scene in the driving primitive domain(s) for a specific point in time. 36, 163, 164
$\mathbf{m}$ A vector representing the signature of a maneuver. xiv, 36, 37, 48–53
$\mathbf{x}$ The (partial) scene for a specific point in time. 13, 14, 17, 36, 37, 61–68
$\mathbf{p}$ The position in a certain coordinate system. 13, 16, 17, 27, 62, 63
$\mathbf{k}$ 36, 37, 43, 45, 47, 63

Functions

$t(\mathbf{k})$ Transform a context into the driving primitive domain such that $t(\mathbf{k}) : \mathbf{x} \rightarrow y$ 36, 37, 47

Sets and sequences

$\mathcal{Y}$ A finite time series representing the evolution of scenes in the driving primitive domain(s) for a period of time. xiv, 36, 37, 47, 49–53
$\mathcal{H}$ A set of scenario hypotheses. 105–107
$\mathcal{I}$ An intersection of a road network. 84, 105
$\mathcal{M}$ The set of maneuvers 36, 37, 47, 48, 50, 51

$\mathcal{K}$ The set of driving primitive contexts to transform a (partial) scene into a the corresponding domain of driving primitive 36, 37, 43, 63

$\mathbb{D}$ The domain of driving primitives xiv, 36–39, 44–46, 51–53, 61, 163, 164

$\mathcal{D}$ The set of driving primitive domains for a use case to represent specific maneuvers 36, 37, 39

$\mathcal{N}$ A road network for a specific geographical area. 84, 88–91, 105

$\mathcal{X}$ A finite time series representing the evolution of scenes for a period of time. 14, 36, 37, 45, 47, 61–63, 66–68, 106, 107

$\mathcal{V}$ The set roads in the proximity of a specific location defined by, e.g., a road user's pose. 61–64, 68

Listings

Chapter 1

Introduction

MOBILITY is a fundamental human need, symbolizing freedom and independence. It facilitates participation in social activities by establishing connections between various central areas of life. [1] In urban areas, the landscape of mobility has evolved in recent years, offering diverse options for individual transport beyond traditional car use and non-motorized travel like cycling and walking. These alternatives include public transportation, car-sharing services, and electric scooters that are readily accessible in larger urban areas.

While mobility grants freedom, it is challenged by the climate crisis. The transportation sector is a significant contributor to greenhouse gas emissions, necessitating urgent reductions to mitigate environmental impact. [2] In 2023, the transportation sector is responsible for 22% of the total greenhouse gas emissions in Germany and is predicted to overshoot the sector's total cumulative emission until 2030 as defined in the German Federal Climate Change Act. [3] One promising approach to achieving the agreement on greenhouse gas emissions is the introduction of electric vehicles (EVs). The technology behind EVs has been under development and testing for many decades, offering the advantages of reducing global and local emissions. [4]

Another technology that represents a major advance in the field of mobility is the emergence of automated vehicles (AVs). The journey towards automated driving began with Bertha Benz's historic drive in August 1888, and future's connected automated vehicle (AV) technology may open new avenues for mobility concepts. Specifically, it holds the potential to provide better mobility for "*underserved individuals such as the elderly population and people with disabilities*" [5], and to significantly lower greenhouse gas emissions of the transportation sector due to increased fuel economy [6] and positive impacts on traffic flow [7]. Moreover, and not less important, it contributes to the Vision Zero initiative, which aims to eliminate traffic fatalities and severe injuries. Over the past decades, driver assistance systems have substantially reduced accident-related fatalities, yet both urban and rural areas continue to witness accidents, some with fatal outcomes.

1.1 Problem definition

However, AV technology faces numerous regulatory and technological challenges. Key issues include determining liability in the event of system failures [8] and addressing ethical dilemmas [9], such as decision-making during unavoidable accidents. Ensuring that AVs can reliably perform automated functions without having a human driver as backup is

a complex task, necessitating sophisticated solutions to handle real-world scenarios. AVs must independently manage tasks such as route planning, maneuver execution (e.g., turning at intersections), and decision-making based on real-time environmental analysis — everything in compliance with local regulations and in an open environment. For these tasks, an AV need to accurately perceive their surroundings and predict the behavior of other road users, especially in future mixed-traffic conditions involving both automated and human-driven vehicles.

1.1.1 Safety validation of automated vehicles

Before AVs can be deployed on public roads at scale, several challenges must be addressed. Chief among these is the development of advanced driving functions that operate sufficiently safe and in compliance with local regulations and laws. [10] Specifically, AVs must not only meet regulatory standards, but also demonstrate safety within their defined operational environments, known as Operational Design Domain (ODD). These domains are specified by manufacturers and delineate the conditions under which an AV can operate safely. [11] Ensuring that AVs behave safely within the ODD is paramount, as these vehicles are expected to navigate the complexities of the real-world with a level of safety that matches or exceeds human driving capabilities. The safety assurance of these systems must be rigorously validated both during development and before deployment.

Traditional distance-based safety verification methods, which rely on accumulating vast amounts of mileage with a Vehicle under Test (VuT) to statistically prove safety, face significant limitations, especially with driving functions of higher automation level. The primary challenge with this approach is the rarity of traffic accidents or critical encounters. This infrequency, as illustrated by the severity pyramid (see Figure 1.1) presented by Hydén *et al.* [12] in 1984, highlights the disproportionate relationship between the severity of events and their frequency in real-world traffic. For instance, traffic accidents, particularly those resulting in fatalities, are among the rarest events but have the highest severity. Wachenfeld *et al.* [13] underscores this point with the example of a highway pilot system, demonstrating that multiple hundreds of million kilometers of driving data would be required to statistically prove safety in Germany, which is economically not feasible.

The world around us is inherently chaotic. Even with the most sophisticated methods for the safety argumentation of AVs, there will always be a chance that the AV will encounter an unforeseeable situation, and may be unable to prevent an accident. This is at first due to the fact that there is no generic situation that leads to an accident and with which the AV can be extensively tested. Sometimes it is due to the chaotic world in which we live that bad things happen. Lin [9] provides an illustrative example of this phenomenon with respect to animals, in particular of deer, crossing the street. An AV may be capable of rapidly interpreting the situation and devising countermeasures to avert a critical incident in a multitude of circumstances. However, this may not be the case in all instances. This example also raises ethical questions pertaining to the intended behavior of AVs to handle such situations. Lin [9] further illustrates this with another scenario with humans involved. In this scenarios, an AV must choose between "*either swerve left and strike an eight-year old girl, or swerve right and strike an 80-year old grandmother*" [9]. Although this

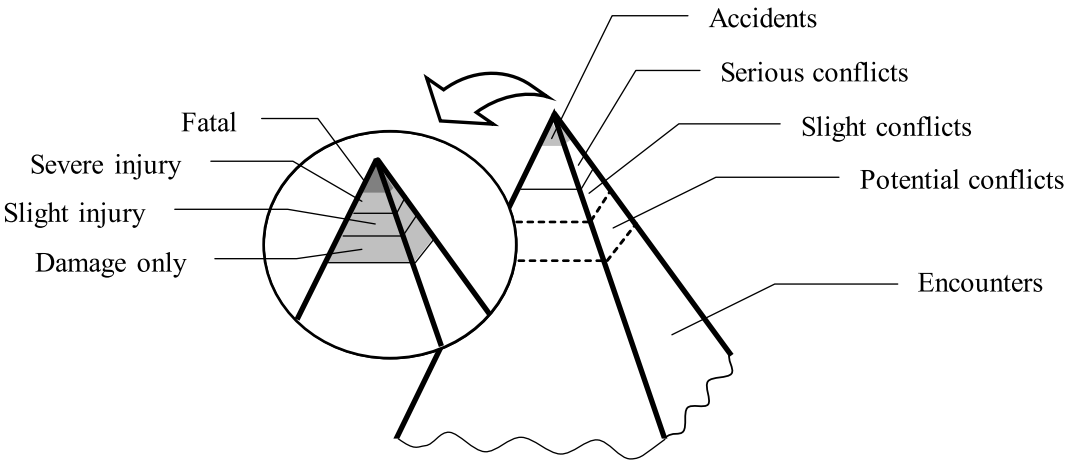


Figure 1.1: The safety pyramid by Laureshyn *et al.* [14] based on Hydén *et al.* [12] to categorize events for interactions between road users according to their degree of severity.

is a terrible choice to make, and even if this is a rare case, an AV may have to make that choice, while the underlying behavior must be implemented and signed-off by humans. [9]

1.1.2 Scenario-based testing of automated vehicles

Two pivotal questions that must be answered by the society, automakers and governments when it comes to the safety argumentation of AVs are thus „How safe is safe enough?“ [13] and „How certain are we that AVs are safe?“ [15]. Wachenfeld *et al.* propose two goals for the validation of AVs in terms of safety, recognizing that the AV technology may come with a potential risk. That is, the individual who uses the technology should be informed that the usage may pose a higher risk on them. It is thus up to the user to decide whether they take the risk, weighing up advantages and disadvantages. Furthermore, releasing the AVs into the public domain should not expose the overall society to a higher risk. [13] The second question is addressed by numerous research projects and initiatives such as Pegasus [16], VVM [17], AdaptIVe [18] and StreetWise [19]. They advocate for a scenario-based approach for the safety argumentation of an AV. scenario-based testing (SBT) shifts the focus from distance to specific, predefined driving scenarios. Scenarios such as making a turn at an urban intersection or merging onto a multi-lane highway encapsulate critical and diverse driving situations.

SBT adheres to the „divide and conquer“ paradigm, an approach that is particularly useful in addressing complex problems. The overarching challenge (the safety argumentation of an AV) is often too intricate and broad to be effectively managed as a whole. Instead, the problem space is decomposed into more manageable subproblems. These subproblems are expected to be simpler and therefore easier to handle compared to the original problem. Each subproblem represents a specific scenario within the broader context defined by the

ODD. By breaking down the complex problem into these distinct scenarios, it becomes possible to focus on the safety validation within the scenarios. Hence, by mapping the ODD to a comprehensive set of these scenarios, the scenario-based approach aims to reduce the verification burden while maintaining rigorous safety standards. These scenarios are then employed as test cases to build robust safety arguments for the AV system. [17]

The overall process for the development of AV typically follows the widely established V-Model and interested readers are referred to [20]. SBT is used throughout this process in different stages. An abstract overview of SBT is given by Neurohr *et al.* [21] in Figure 1.2. The blue boxes denote artifacts along the process, while the black rectangles represent specific tasks. The overall process can be decomposed into three consecutive steps, although this is a simplified illustration. The first step is to define the scenarios that helps in the safety argumentation. Furthermore, the system requirements need to be defined, such as the overall safety goal. Both steps provide artifacts for the actual testing of the AV. The scenarios and the requirements are utilized for the derivation of test cases. The methods that are applied for testing depend on the development stage and include virtual, real-world or a combination of both (refer to Huang *et al.* [22] and [23] for an overview of testing methods). What follows after the conduction of the tests is the analysis of the test results w.r.t the initially defined requirements. The testing results are in the last step used as part of the overall safety argumentation of the AV. This work will mainly focus on the first part of this process, the identification of scenarios in real-world traffic data.

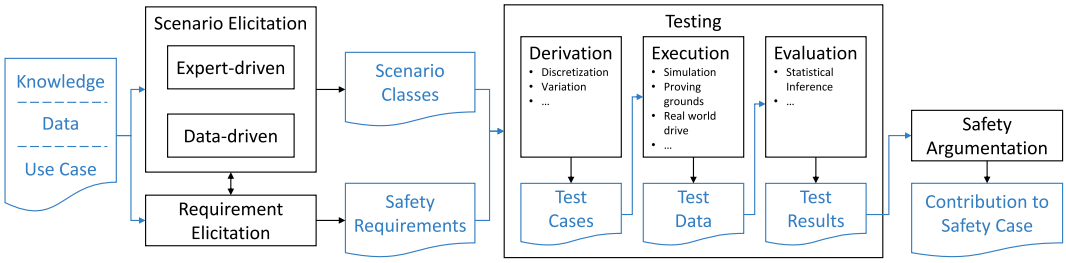


Figure 1.2: An abstract overview of scenario-based testing by Neurohr *et al.* [21]

1.1.3 Traffic data sources for real-world scenarios

The adoption of SBT has revealed the utility of a comprehensive scenario database in ensuring the safety of AVs. [24] [25] Over recent years, various methodologies have been explored to define these scenarios. These methodologies are typically categorized into two domains: expert knowledge or data-driven. [21] The latter includes accident datasets with varying severities, synthetic data sources using simulation environments like CARLA [26][27], real-world traffic data using floating vehicles [28][29] or stationary infrastructure, or a combination of both. [21]

Regardless of the source, the common goal of these approaches is to identify scenarios. They can be used to create specific test cases, which are crucial for validating AVs. Moreover, multiple similar scenarios allow the investigation of human driving behavior,

which helps in the development of AVs. Therefore, the following section provides a brief overview of the three typical methods for collecting traffic data in the real world.

Floating vehicle One effective method for acquiring traffic data in the real-world is through the use of floating vehicles. This approach involves equipping vehicles with sensors to record Naturalistic Driving Data (NDD), providing valuable insights into traffic dynamics and vehicle behavior.

Several datasets have been made publicly available using floating vehicles. Notable examples include the Waymo Motion Dataset [30], which encompasses 500 hours of driving data across 70,000 scenarios, and the nuPlan Dataset [31], featuring 1,200 hours of data collected from four different cities. The datasets have generated significant interest within the research community, and they have been utilized to address a diverse array of problems, including object detection and tracking [32] and scene understanding [33].

To capture both the surrounding environment and the vehicle's own state, floating vehicles are typically outfitted with a variety of sensors. Cameras, LiDAR, and RADAR sensors are utilized to gather detailed information about the vehicle's surroundings, enabling precise environmental mapping and object detection. Positional sensors such as the Global Navigation Satellite System (GNSS) and Inertial Navigation Systems (INS) provide accurate vehicle positioning and motion data.

The utilization of floating vehicles for traffic data collection offers several significant advantages. These vehicles can record traffic information across different regions, providing a comprehensive overview of traffic patterns and behaviors in diverse environments. The integration of advanced sensors ensures high precision in both environmental perception and self-positioning, enhancing the reliability of the collected data. However, the trajectories of other road users are incomplete, which can limit the scope of traffic analysis. Additionally, the extent of traffic information collected is constrained by the size of the vehicle fleets. Larger fleets are necessary to provide continuous and comprehensive temporal data coverage, which can be resource-intensive.

Quasi-stationary infrastructure A different method for collecting traffic information involves quasi-stationary installations, such as drones and mobile masts as illustrated in Figure 1.3. Unlike vehicles, these methods provide trajectories of traffic participants across the entire coverage area. This capability enables more detailed analyses regarding traffic safety.

Drones, such as the one illustrated in the left panel of Figure 1.3, represent a relatively novel methodology for collecting traffic data in real-world settings, first being utilized for this purpose in 2016 by Robicquet *et al.* [34]. Since then, multiple datasets have been published with different scopes and applications. Notable examples include the highD dataset [35], which focuses on highways; the inD dataset [36], which captures data from German intersections; and the openDD dataset [37], which is dedicated to roundabouts; and the exiD dataset [38], which focus on highway entries and exits.

The recent gain in popularity of drones for traffic data acquisition are due to technological advances in that field since they serve as a flexible and cost-efficient method for traffic

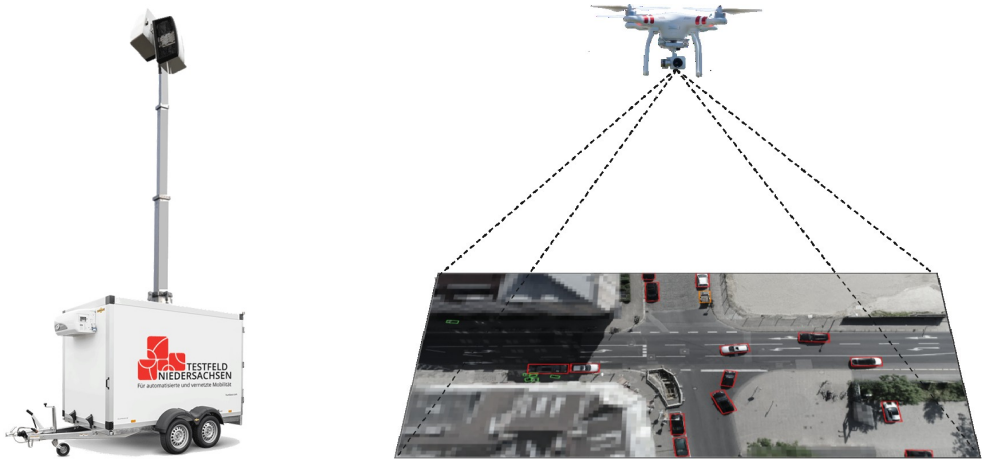


Figure 1.3: Quasi-stationary methods for traffic data acquisition. *Left:* A mobile measurement station (© DLR e.V. [39]). *Right:* A drone hovering over a measurement site [40].

data acquisition. In fact, drones offer flexibility in terms of observation location. This mobility allows for the collection of traffic data from diverse and challenging environments. Moreover, the bird's-eye view provided by drones eliminates the issue of occlusion by road users, ensuring a clear and unobstructed field of observation. [38] Furthermore, the high-resolution cameras equipped on drones enable precise and accurate object detection [40], which is crucial for detailed traffic analysis. Finally, due to their capability to hover at elevated heights, drones do not interfere with or alter human driving behavior, thereby ensuring the authenticity of the captured data. [40]

The utilization of portable masts and drones differ fundamentally from traditional vehicle-based data collection. While vehicle-based systems are limited to the paths taken by the vehicles themselves, quasi-stationary installations can capture the movements of all traffic participants within their field of view. This comprehensive data collection approach is particularly valuable for traffic safety analysis.

Stationary infrastructure In recent years, the collection of traffic data through stationary infrastructure has gained significant traction, particularly within the context of test beds in Germany. In contrast to quasi-stationary infrastructure, the sensors are permanently mounted for long-term analysis. This is done either at existing infrastructure — a method long established — or at newly installed poles. This has become a cornerstone in the realization of advanced traffic monitoring and autonomous driving test sites, such as the Ted Bed Lower Saxony Niedersachsen [39] and Test Area Autonomous Driving Baden-Württemberg [41].

Stationary methods employ a variety of technologies, including cameras, LiDAR, and RADAR, sometimes in combination. To ensure continuous data collection using camera-based systems even under low-light conditions, they sometimes incorporate active in-

frared flashes. [42] This multifaceted approach enables an accurate representation of traffic conditions, similar to the quasi-stationary methods.

One significant challenge encountered with stationary sensors is occlusion. That is, larger road user close to the camera could occlude the view to other smaller road users behind them. To mitigate this issue, sensors are typically installed on multiple masts with different angles of view, creating overlapping coverage areas. This configuration ensures a more robust and reliable capture of traffic dynamics.

While stationary sensors provide a geographically limited data collection scope, they offer the advantage of continuous monitoring over extended periods. This capability allows for the observation of traffic under various weather conditions and at different times of the day, facilitating the identification of rare traffic phenomena. This long-term, uninterrupted data collection by stationary sensors is invaluable for comprehensive traffic analysis and the safety assessment of AV technologies.

A comprehensive overview of three approaches to traffic data collection is shown in Table 1.1. The evaluation is based on the criteria: temporal coverage, spatial coverage, environment model accuracy, and environment model comprehensiveness. Each method is rated on a scale from "–" (very poor) to "++" (excellent) in these categories based on the description above. The table shows the strength and weaknesses of each approach, while the ranking is based on each method's description from above.

Table 1.1: Overview and evaluation of real-world traffic data collection methods.

Type	**Coverage**		**Environment Model**	
	Temporal	**Spatial**	**Accuracy**	**Comprehensiveness**
Floating vehicle	−	++	++	*o*
Quasi-stationary infrastructure	+(*o*)	*o*	+	++
Stationary infrastructure	++	−−	+	++

Floating vehicle data excels in spatial coverage and accuracy of the environment model. They allow for extensive geographic data collection, capturing real-time traffic patterns and road conditions with high precision. However, this method lacks in temporal coverage since data collection is limited to the periods when the vehicles are in motion — although this depends on the fleet size.

Quasi-stationary infrastructure offers a balanced approach. It provides a high comprehensiveness of the environment model by employing drones or multiple masts. The bird's-eye view from drones or the diverse angles from multiple masts enable detailed traffic monitoring and analysis. Nevertheless, quasi-stationary infrastructure is constrained in spatial coverage compared to floating vehicles, as it is typically deployed to monitor traffic within specific regions rather than over extensive areas.

Stationary infrastructure provides extensive temporal coverage since it allows continuous traffic data collection. Similar to quasi-stationary infrastructure, it also achieves a high degree of comprehensiveness in the environment model. However, stationary infrastructure has significant spatial limitations. Their main purpose is to monitor traffic in a specific area and relocating them to another region is either not possible or very resource intensive.

The table reveals that each traffic data collection method has its unique strengths and weaknesses. The selection of an appropriate methodology depends on the specific requirements of the task at hand, highlighting the importance of considering the trade-offs between temporal coverage, spatial coverage, and environment model comprehensiveness. Floating vehicles may be used for the identification of scenarios in various environments. Quasi-stationary infrastructure for the identification of scenarios in multiple regions with information of all traffic participants. Stationary infrastructure for long-term traffic data collection of all traffic participants that enables in-depth scenario analysis.

1.2 Thesis objectives

While the manual effort to identify relevant scenarios in individual test drives of VuT may still be economically feasible, it becomes impractical as the size of vehicle fleets increases, as during Naturalistic Driving Study (NDS) or Field Operations Test (FOT). This is particularly true for the identification and extraction of scenarios from data collected using stationary infrastructure, as multiple scenarios can occur simultaneously and data is continuously gathered. This continuous stream of data poses a significant challenge, necessitating suitable technical infrastructure and processes to address it. Consequently, the primary research question of this thesis is:

Research Question I *How can a framework be defined that allows for the systematic representation of continuously collected real-world traffic data acquired by (quasi-) stationary infrastructure in terms of scenarios?* **RQ.I**

As described in the last section on the various methods for collection real-world traffic data, floating vehicles are equipped with sensors capable of accurately capturing their immediate surroundings. Under this assumption, the aforementioned research question also implies that this framework should enable the processing of traffic data collected by vehicles. Throughout this dissertation, traffic data collected by vehicles and traffic infrastructure will be utilized, with the focus primarily on the latter due to the defined requirements. Based on the **RQ.I**, the following objective is derived:

Objective I *Development and evaluation of an approach that allows the systematic identification of scenarios from real-world traffic data that is collected from (quasi-) stationary infrastructure.* **O.I**

For the scenario-based representation of traffic data, a procedure is necessary that systematically enables this. This involves both defining a specific process and developing a methodology to systematically define scenarios and extract them from real-world traffic data. The proposed approach must be discussed and evaluated using a concrete problem to demonstrate its practical applicability. This scenario-based representation procedure should be integrated into the overall framework for the continuous identification of scenarios. From this, and in relation to RQ.I, the following additional objective can be defined:

Objective II *Conceptualization and development of a scalable and modular platform that allows to continuously and systematically identify, extract, and analyze scenarios from real-world traffic data collected using (quasi-) stationary infrastructure.* **O.II**

As described in the previous section, quasi-stationary infrastructure allows for the recording of traffic data at various locations over extended periods. One use case is measurement campaigns, where data analysis occurs either during the recording period or subsequently. Depending on the specific use case or the questions being addressed, particularly concerning the analysis of certain traffic phenomena, different scenarios become relevant. For these purposes, suitable flexible structures must be defined that allow methods to be integrated into the overall architecture to address the specific problems or questions that may only be relevant to specific campaigns. This also applies to situations where stationary infrastructure is used to collect traffic data. The platform to be developed must be scalable to enable the continuous processing of traffic data flow and sufficiently flexible to be adapted to new questions and challenges that may arise. Specifically, the adaptability of the platform to new issues and problems should be demonstrated through concrete real-world applications.

For the identification of scenarios in real traffic data, these scenarios must first be described, and this description must then be translated into methods that can identify concrete instances of the scenario in the data. Scenarios are typically described initially using expert knowledge on a semantic level using human language, offering the significant advantage of being understandable and definable by a broad range of stakeholders. An exemplary abstract description of a scenario might be: "A car turns left at an urban intersection while other vehicles are approaching from the opposite direction." This describes a scenario involving a so-called "ego vehicle" intending to turn left at an intersection, encountering other traffic participants, the so-called "challengers" passing through the intersection in the opposite direction. [1] For the identification of this scenario in real-world traffic data, it is beneficial to map the traffic information, which is collected sensor-based as mentioned earlier, onto a semantic level and thus closer to the initial scenario definition.

[1] The term "ego" and "challenger" is used in this work as proposed by Weber *et al.* [43], with the "ego" being the traffic participant of main concern and the others being the "challengers". If traffic data is collected using floating vehicles, the VuT is typically the "ego".

In fact, in defining a scenario, the emphasis is usually placed on modeling the dynamic behavior of traffic participants and the potential interactions among them and the infrastructure. A common approach for scenario identification involves detecting certain maneuvers, such as turning at an intersection or overtaking another traffic participant, and inferring the scenario based on this information. Various methods are proposed for maneuver detection, which either consider maneuvers in isolation or utilize a pre-defined catalog of maneuvers based on expert knowledge. However, maneuvers differ in complexity from both the control perspective of automated vehicles and the data-driven perspective of an external observer. For instance, describing the maneuver of overtaking a vehicle on a highway or a cyclist on a country road requires a higher level of detail and more information than the maneuver of braking, which only requires information about the vehicle's dynamics. One approach to manage the different complexities of maneuvers is, again, the utilization of the „divide and conquer" paradigm, where traffic data are described not only based on events and maneuvers, but also on so-called primitives. [44] [45] [46] The goal is to describe complex driving behavior using simple, smaller building blocks, which can then be used to describe more complex maneuvers or to analyze driving behavior. [46] To achieve this, relevant primitives can be specifically searched for in the traffic data. Methods of unsupervised learning are typically used to structure traffic data with primitives, which requires expert knowledge to determine what phenomenon a primitive represents. [46] This is problematic when these primitives are to be used as the basis for maneuver description. Additionally, it complicates the mapping of phenomena identified in real-world data onto specific terms in the scenario description. This leads to the following research question:

Research Question II *How can real-world traffic data collected by (quasi-) stationary infrastructure be semantically represented using primitives? Furthermore, how can these primitives be utilized to derive maneuvers and identify them in traffic data?* **RQ.II**

Based on the research question, two objectives are derived.

Objective III *Development of a generic methodology that allows the representation of real-world traffic data as primitives and the definition of maneuvers based on them.* **O.III**

The methodology to be developed aims to provide a framework not only for representing the dynamic behavior of traffic participants using primitives, but also for describing phenomena in the environment, which are specified on a semantic level in a scenario description, using primitives. This includes dynamic components such as maneuvers, as well as relationships between traffic participants and between traffic participants and infrastructure. By translating traffic data to a semantic level consisting of primitives, it becomes possible to specifically search for these phenomena and describe more complex phenomena, such as maneuvers, based on these primitives. The framework is intended to serve as

a guideline rather than to dictate specific methods. This allows the selection of methods based on the available real-world data and the phenomena to be described.

Another objective can be derived from the second part of the RQ.II, which is only partially addressed by O.III and defined as follows:

Objective IV *Demonstration of the methodology for representing traffic data using primitives, focusing on specific challenges such as identifying and extracting maneuvers from real-world data. The methodology should be exemplified through case studies addressing these challenges.* **O.IV**

While the goal of O.III focuses on defining the methodology for representing traffic data in the form of primitives, the overarching goal of O.IV is to demonstrate the application of the methodology to solve concrete problems using case studies. It should be exemplarily shown how primitives can be defined. In addition, it is to be shown how properties and states of traffic participants and relations between them and between traffic participants and other entities such as infrastructure can be converted into primitives on the basis of real-world data. Furthermore, it should be demonstrated how this representation with primitives can be utilized, for instance, to describe and identify specific maneuvers. Specifically, the aim is to apply the methodology to solve specific problems that arise during the conversion of real data into scenarios. The applications, the solution approach and its evaluation are to be examined and discussed in the context of case studies with real traffic data.

1.3 Thesis structure

The remainder of this work is structured as follows.

In Chapter 2, basic terms that are used throughout the work are defined. The following three chapters cover topics along the process from traffic data to scenarios, while proposing methods for different problems. Chapter 3 presents a methodology to represent traffic data semantically using primitives. The primitives are the building blocks that are utilized for various use-cases, including the definition of maneuvers and the semantic representation of relationships between traffic participants.

One of the main tasks to extract scenarios from real-world traffic data is to associate traffic participants on specific roads and, especially on intersections, identify driving maneuvers. This topic is covered in Chapter 4 which presents methods for driving maneuver identification that is based on an a priori enriched digital representation of the road network in OpenDRIVE format. In both previous chapters, methods are described that transform and enrich traffic data. Chapter 5 follows with the utilization of these information and presents an approach to build a knowledge base of scenarios. Specifically, the chapter shows how to combine a knowledge-based approach for the definition of so-called space-sharing conflict scenario (SSCS) hypotheses with a data-driven approach that is used to validate the scenario hypotheses.

The work follows with the presentation of the overall framework to extract scenarios from traffic data in Chapter 6. The chapters elaborate on details of such a platform, beginning with the architecture. Additionally, the overall process of extracting scenarios from real-world traffic data is discussed, and it is shown how that process is transferred to an architecture for a flexible and scalable processing pipeline. The chapter also presents components for traffic data management, processing and analysis. Finally, the chapter concludes with a demonstration of the framework's flexibility by the application in different use-cases.

In Chapter 7, the thesis concludes with a summary of this work and a discussion of the its contributions. Furthermore, future research directions are outlined.

Chapter 2

Foundation and related work

This chapter introduces basic terms and concepts that are fundamental for the remainder of the work. In particular, Section 2.1 briefly describes trajectory data. Furthermore, foundations regarding the representation of traffic data (Section 2.2) and in particular the scenario-based representation of traffic data (Section 2.2.3) are presented. Lastly, this chapter revisits metrics for the situational assessment of traffic safety in Section 2.4. It is worth noting that this chapter only covers foundations that are fundamental for this work, while those that are only relevant for specific applications are briefly described in the corresponding chapters for the sake of clarity.

2.1 Basic terminology

This work aims to represent real-world traffic data through scenarios. Traffic data encompasses various types of information (see Chapter 3), with geospatial details about traffic participants and their dynamics, such as speed or acceleration, being essential for describing traffic comprehensively. The following section briefly introduces the basic terms used throughout this work and related works related to scenario-based representation of traffic data.

2.1.1 Pose, position, trajectory

Analyzing the driving behavior of traffic participants in specific situations requires information about where the traffic participants are. The **position** of traffic participants is typically defined by the coordinate and the geodetic reference system (see Section 2.1.2). [47] In this work, the coordinate $\mathbf{p}$ is defined as

$$\mathbf{p}_t = \begin{bmatrix} p_1 & p_2 \end{bmatrix}^T \tag{2.1}$$

in a two-dimensional reference frame at a specific time t, unless stated otherwise.

The position is generally combined with information about the traffic participant's dynamics and absolute orientation. The **pose** $\mathbf{x}$ of a traffic participant includes these details and is defined as

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{p} & v & a & \omega \end{bmatrix}^T \tag{2.2}$$

where v is the velocity in $\frac{m}{s}$, a is the acceleration in $\frac{m}{s^2}$ and ω is the absolute orientation in radians for time t, unless stated otherwise. Note that in some parts of this work, the

pose may only include the position and the orientation, or additionally contains the traffic participant's dimension, which will be explicitly stated.

The position and pose of a traffic participant describe its state for a specific time t. Vehicles and intelligent traffic infrastructure typically collect information continuously. The **trajectory** of a traffic participant represents its evolution over time. Let $\mathcal{X}$ denote the trajectory represented as a time-series with a fixed duration T, it is defined as

$$\mathcal{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T) \tag{2.3}$$

with $\mathbf{x}_t$ representing the pose of that traffic participant at time t. Figure 2.1 shows an example of a trajectory acquired by the stationary infrastructure AIM Research Intersection of the DLR. The trajectory denotes the overall course of the traffic participant within the measurement area and is represented as the blue line. The pose represents it's state in a specific point in time, while the dots and the arrows illustrate the positions and orientations.

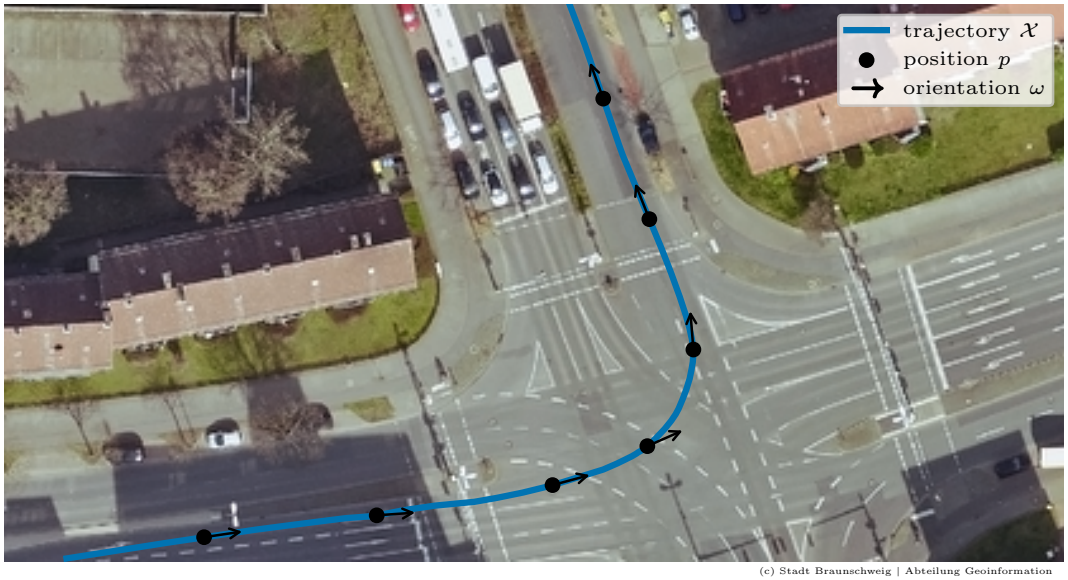


Figure 2.1: A trajectory on the DLR AIM Research Intersection.

2.1.2 Coordinate systems and reference frames

So far, basic concepts were introduced, including the definition of a trajectory and position. In Section 2.1.1, the position is defined by coordinates and a geodetic reference system. When traffic data is acquired by vehicles, quasi-stationary infrastructure or static infrastructure, different coordinate systems and reference frames are typically utilized.

The reference system or reference frame is mandatory to uniquely specify a position since it specifies the position of a coordinate system relative to the earth. [47] The coordinate system defines how to interpret the numerical values of positions, with geographic

coordinates (latitude, longitude and height) and Cartesian coordinates being the two most used variants of coordinate systems. While the ISO Norm 19111:2019 "*defines the conceptual schema for the description of referencing by coordinates*" [48], a brief overview of relevant systems is given in the following.

World Geodetic System (WGS)

The World Geodetic System (WGS) is one of many geodetic reference systems that approximate the earth by ellipsoids. [47] Its latest realization from 1984 is, for instance, used by the Global Navigation Satellite System (GNSS) Global Positioning System (GPS) [47] and in many applications such as navigation, mapping and geospatial analysis.

As illustrated in Figure 2.2, it provides a good localization accuracy on the earth's surface. However, other ellipsoids or reference frames are typically utilized in regional applications that fits better to the actual surface, such as the European Terrestrial Reference System 1989 (ETRS89) for positions in Europe.

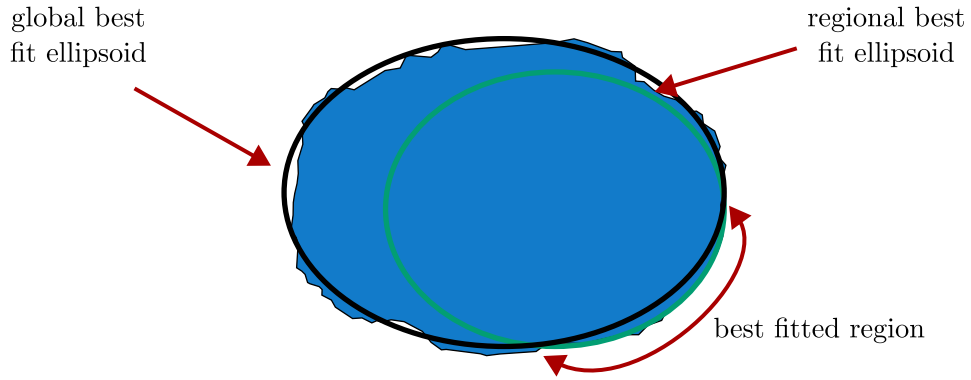


Figure 2.2: The earth is approximated by various ellipsoids dependent on the use-case (based on Lange [47]).

Universal Transverse Mercator (UTM)

The UTM coordinate system is a Cartesian coordinate system and usually uses the WGS84 reference system. A position in the UTM coordinate system is given in easting and northing and derived by projecting geographic coordinates onto a plane. [47] To this end, the earth is divided into 60 vertical zones, each with a longitudinal extent of six degrees, and positions on the earth's surface are projected on a cylinder that is wrapped around it, opened and flattened (see Figure 2.3). [49] The projection of positions on a plane surface has several benefits, including the distance estimation using the Euclidean distance and the visualization of trajectories.

Modern vehicles are usually equipped with GNSS-based positioning systems and combine it with an inertial navigation system. [51] Therefore, the position is typically represented in geographic coordinates and in a global reference system such as WGS84. But, stationary intelligent traffic infrastructure usually acquire traffic data in a small region and

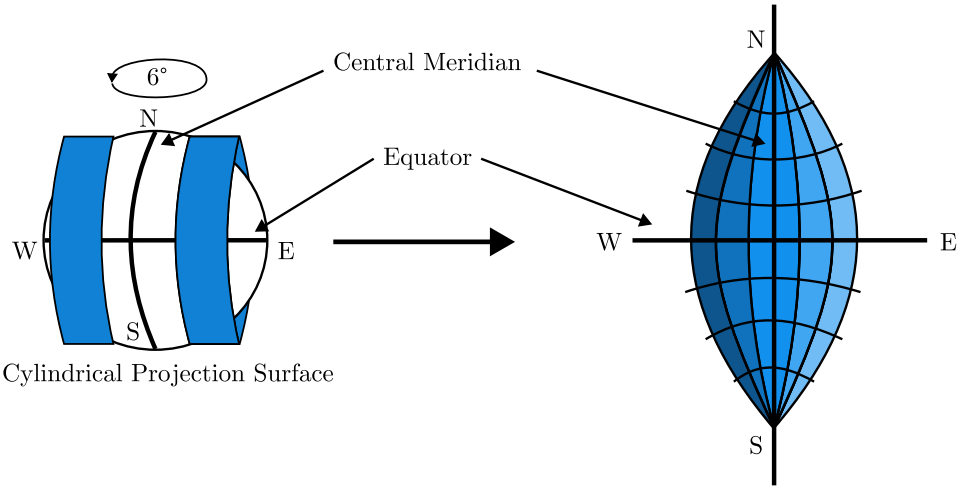


Figure 2.3: The transverse mercator projection using a cylindrical projection surface around the earth's surface (based on Olsen [50]).

therefore represent positions in a Cartesian coordinate system such as UTM. To fuse positions acquired by the vehicle and by intelligent traffic infrastructure, and for the analysis of trajectory data, the positions throughout the work are represented in UTM coordinates such that (2.1) is redefined as

$$\mathbf{p}_t = \begin{bmatrix} u & v \end{bmatrix}^T \tag{2.4}$$

with u, v denoting the position in easting and northing. For instance, the AIM Research Intersection illustrated in Figure 2.4 acquires traffic data within the area (604665, 5792720), (604865, 5792858) and thus covers an area of approximately 200 times 150 meters.

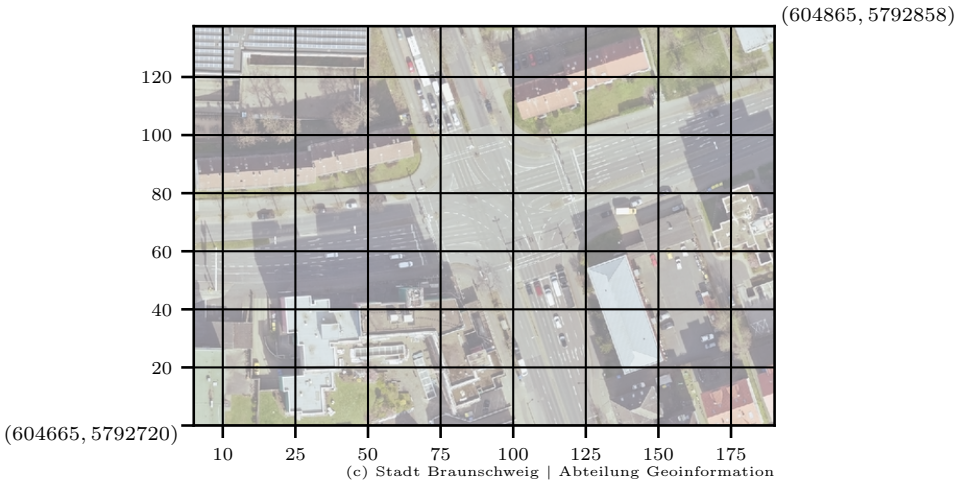


Figure 2.4: The DLR AIM research intersection area represented in the UTM coordinate system.

Road reference line

The UTM coordinate system allows to represent positions in a fixed reference frame. Another coordinate system that is used throughout this work employs a dynamic reference frame, the road reference line coordinate system. Other than the UTM coordinate system, the road reference system uses the digital representation of the road as reference for positions. This coordinate system is used in the ASAM OpenDRIVE standard, which is a file format specification for road network descriptions. The road reference line coordinate system is related to the Frenet Frame, which is a moving frame along a curve [52] that is, for instance, utilized for motion planning of automated vehicles [53].

The position $\mathbf{p}_t$ at time t represented in a Cartesian coordinate system such as UTM is converted into a dynamic reference frame specified by the lane boundary's representation r according to Werling *et al.* [53] by

$$\mathbf{p}_t = \begin{bmatrix} f(\mathbf{p}_t, r) & g(\mathbf{p}_t, r) \end{bmatrix}^T \tag{2.5}$$

with $f(\mathbf{p}_t, r)$ the offset along the road r and $g(\mathbf{p}_t, r)$ the lateral offset relative to the road. Figure 2.5 illustrates an example of a vehicle merging onto the highway. The two highlighted positions (black dots) represent positions in the UTM coordinate system. The positions $\mathbf{x}_{t_1}, \mathbf{x}_{t_2}$ are represented in the Frenet Frame defined by the left border of the acceleration using (2.5). f represents the offset relative to the road's origin illustrated as the vertical left line on the left side. g denotes the orthogonal distance to the road marking. The projection of the absolute UTM coordinates into the local dynamic reference frame allows to reason about the location of traffic participants relative to the road. This not only enables for motion planning [53], but also to associate traffic participants on road lanes, or identify maneuvers. Both use-cases will be demonstrated in Section 3.

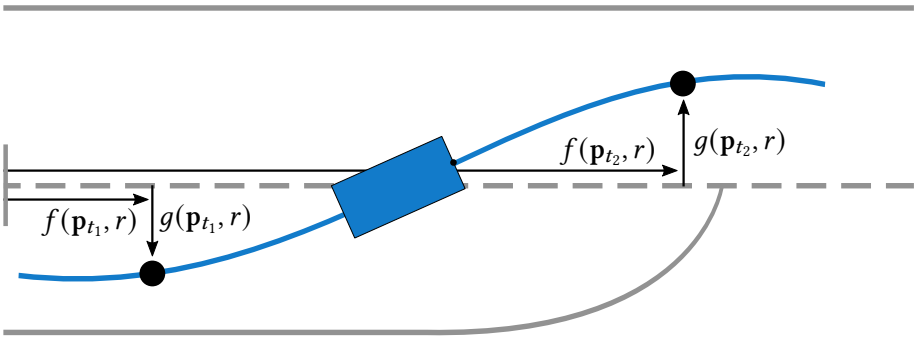


Figure 2.5: The trajectory in the dynamic road reference frame defined by the left border r of the acceleration lane.

2.1.3 Scene, primitive, maneuver, scenario

In the domain of scenario-based representation of traffic data, several concepts are widely used to describe specific aspects of traffic data. This work employs a hierarchical view on

traffic data using a four layered model as illustrated in Figure 2.6. In the following, relevant terms that are also used throughout this work are briefly introduced.

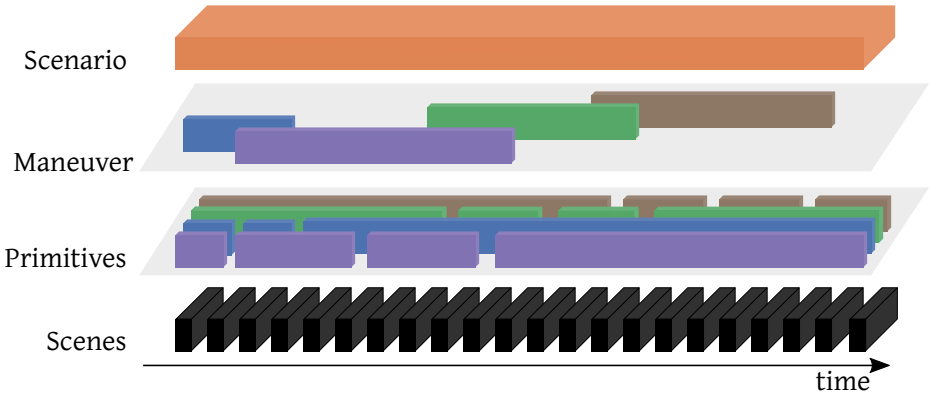


Figure 2.6: The four layered model for a scenario-based representation of traffic data used throughout this work (based on [54]).

Traffic data include various types of information and typically represents traffic quantitatively in a fixed spatial area and over a certain period of time. The term **scene** relates to a subset of the overall traffic data. [55] [56] While the definition of a scene from Geyer *et al.* is ambiguous w.r.t the duration of a scene, Ulbrich *et al.* uses the concept of a snapshot of the overall traffic data, such that a scene represents traffic data for a specific point in time. In general, this work follows the scene definition of Ulbrich *et al.*, while the restriction that a scene does not span "*a certain amount of time*" [56] is relaxed. Traffic data used throughout this work is acquired in the real world by various types of sensors that usually have different sampling frequencies. Thus, a scene is a representation of the aggregated information provided by the sensors for a small period of time, which is dependent on the use-case.

At scene level, information about traffic participants and the environment is represented quantitatively. For instance, the speed of a traffic participants is typically explicitly stated in meters per second or kilometers per hour. To semantically represent traffic data, those quantitative attributes are mapped to a semantic taxonomy. While maneuvers are a widely adopted concept for this task, this work uses **primitives** as an intermediate level between maneuvers and scenes. Other than a scene that represents traffic data for a short period of time, a primitive may last over multiple scenes. Moreover, a primitive represents the state of a traffic participant or other entities involved, and the relations among them. This is illustrated in Figure 2.6 with the colorized boxes along the third dimension of the primitive layer. Every primitive category represents such a relationship among entities (see Chapter 3). In fact, the hierarchical view for scenario-based representation of traffic data illustrated in Figure 2.6 is analogous to the representation of a mission or rather mission element proposed by Dickmanns [57] in 2007. According to Dickmanns, a mission describes the process of driving from a starting position to an arbitrary destination. The mission can be decomposed into mission elements, which denote specific phases of the mission, such as traveling a certain distance or turning right to merge onto a major urban

road with three lanes. Mission elements are further decomposed into maneuver, which are in turn described by maneuver elements. The latter are the basic elements on control level. From a data and scenario-driven perspective, the abstraction level of maneuver elements is the primitive layer used in this work. Wang *et al.* [45] also use the concept of primitives as the fundamental components to represent and generate new traffic scenarios, although the results provided by their presented framework for primitive identification lacks interpretability. But, the unambiguous semantical representation provided by primitives is mandatory since it helps to understand even complex situations. [57]

The scene represents traffic data for a specific point in time. Primitives may span a wider time range and represent traffic data on a semantical level. If we focus on specific traffic participants within the overall traffic data and their behavior, scenes and primitives can be composed into **maneuvers**. That is, traffic participants perform maneuvers in order to travel from their origin to the desired destination. This is formulated by Dickmanns as "*special expressions for these capabilities of motion control*" [57]. Other than scenes that span a short period of time, maneuver "*have a temporal extension in the seconds-to-minute range*" [57].

Following the hierarchical view on traffic data, the most abstract concept to describe traffic data throughout this work are **scenarios**. Due to the fact that SBT gained acceptance by industry and academia, various definition of the term scenario exists as outlined by Gelder *et al.* [58]. In general, this work employs the definition of Ulbrich et al. stating that a "*scenario describes the temporal development between several scenes in a sequence of scenes. Every scenario starts with an initial scene. Actions & events as well as goals & values may be specified to characterize this temporal development in a scenario. [...] [A] scenario spans a certain amount of time.*" [56] Note that as pointed out by Gelder *et al.*, the term event is not defined by Ulbrich *et al.* In this work, the term event is interpreted as by Gelder *et al.* denoting a transition between states of traffic participants or even infrastructure induced by reaching a specific threshold. That is, an event represents the moment in which a transition occurs between any of the three abstraction levels: primitive, maneuver and scenario. However, in the remainder of this work, events are not modelled explicitly since they can be derived from the representation using primitives, maneuvers and scenarios.

The hierarchical representation of the data model depicted in Figure 2.6 illustrates the various temporal progressions of each layer. An alternative representation of the layer model is shown in Figure 2.7. In this figure, the four different layers are depicted, but instead of showing temporal progression, the quantity of elements per layer is illustrated as in the safety pyramid of Hydén *et al.* The number of elements per layer decreases from the lowest level to the highest level, in accordance with the principles of hierarchical data aggregation. Unlike the pyramid of Hydén *et al.*, however, there is no continuous transition between the different layers; instead, defined boundaries exist. This means that traffic information at the scene level is aggregated to primitives at the next level, which in turn allow maneuvers to be defined, from which scenarios can be constructed. Typically, instances of scenarios are categorized based on specific questions, such as traffic safety through interactions between traffic participants. Consequently, the safety pyramid of Hydén *et al.* could be integrated into the scenario level to categorize scenarios ranging

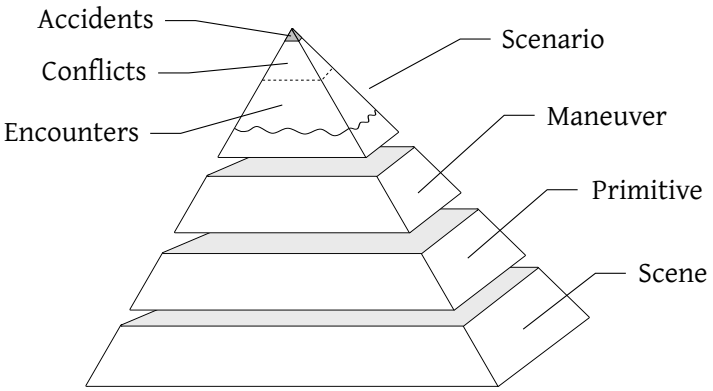


Figure 2.7: The representation of traffic data in terms of scenes, primitives, maneuvers and scenarios utilized in this work.

from simple encounters between traffic participants to scenarios involving accidents of increasing degree of severity.

2.2 Traffic data models

So far, various terms are defined for describing traffic data on a semantical level. In the following, an overview of relevant work is given that propose concepts to structure the different levels of the hierarchical model.

2.2.1 Scene description models

The most detailed description of traffic along the hierarchical model is available on the scene level. In recent years, several works proposed to describe traffic and its environment using models with different level of detail and different perspectives. A general description was proposed by Bagschik *et al.* [59] with a five-level layer model (see Figure 2.8), which is based on the work of Schuldt [60]. Every level of this model represents specific aspects of the scene. The first level is the *Road level*, which contains information about the road's layout such as the topology and geometry. The second level adds information about the *Traffic Infrastructure* and includes traffic signs, among others. While this layer includes information about the infrastructure that is installed permanently or for a longer period, any "[t]emporary manipulations of the first two layers are represented in the third layer" [59]. Specifically, this layer includes information about temporary construction sites and changes to the original road layout. Any objects that are part of the scenery are described in the third level *Objects* including their interactions and their performed maneuvers. According to Schuldt this also includes the control of traffic lights, as a dynamic element, while the physical representation of the traffic light is modelled on the *Traffic Infrastructure* level. For the safety argumentation of AVs it is crucial to know whether the system is able to drive over the full day or only under specific lightning conditions. last level focuses on

the representation of the *Environment*, e.g., weather information, lighting. This representation was superseded by the six-level model introduced by the PEGASUS research project. The additional sixth layer represents digital information for, e.g., vehicle-to-vehicle communication.

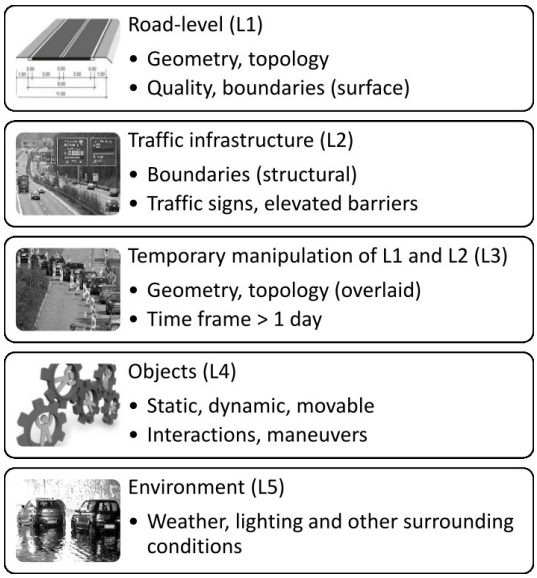


Figure 2.8: The five-level layered model for scene description by Bagschick *et. al.* [59].

In order to ensure the safety of AVs, it is essential to evaluate the effectiveness of the system in all environmental conditions and to identify external factors or phenomena that could potentially push the system to its limits. This evaluation should encompass a range of times of day with varying light conditions and sun positions. For instance, glare from other traffic participants at night could affect the vehicle's sensors, thereby reducing the quality of the environment perception. Furthermore, particular meteorological occurrences, such as precipitation in conjunction with a low sun angle, could result in reflections on the road surface, thereby influencing the vehicle's environment perception. These specifics can be described at the *Environment* level.

2.2.2 Maneuver description models

Instead of this overall representation of traffic data, Hartjen *et al.* [61] focus on a maneuver-driven description using a three-level layered model for the urban domain (see Figure 2.9). The first layer describes the dynamic behavior of a traffic participant with a set of maneuvers, including *Accelerate*, *Keep Velocity*, *Decelerate* and *Reversing*. Specifically, this layer describes the movement of a traffic participants.

The other two layers describe the dynamic relation between a traffic participant and other entities. In the second layer Hartjen *et al.* defined several infrastructure-related maneuvers, such as *Lane change*, *Cross Junction*. Those maneuvers can be utilized to represent

the dynamic behavior of a traffic participant relative to infrastructure, such as crosswalk, junctions and lanes.

The third layer contains the inter-object related maneuvers such as *Follow Object*, *Approach Object*. Effectively Hartjen *et al.* provide a detailed description of the *maneuvers* and *intersections* for the urban domain that are represented in the *Objects* level in the five-level model proposed by Bagschik *et al.* illustrated in Figure 2.8.

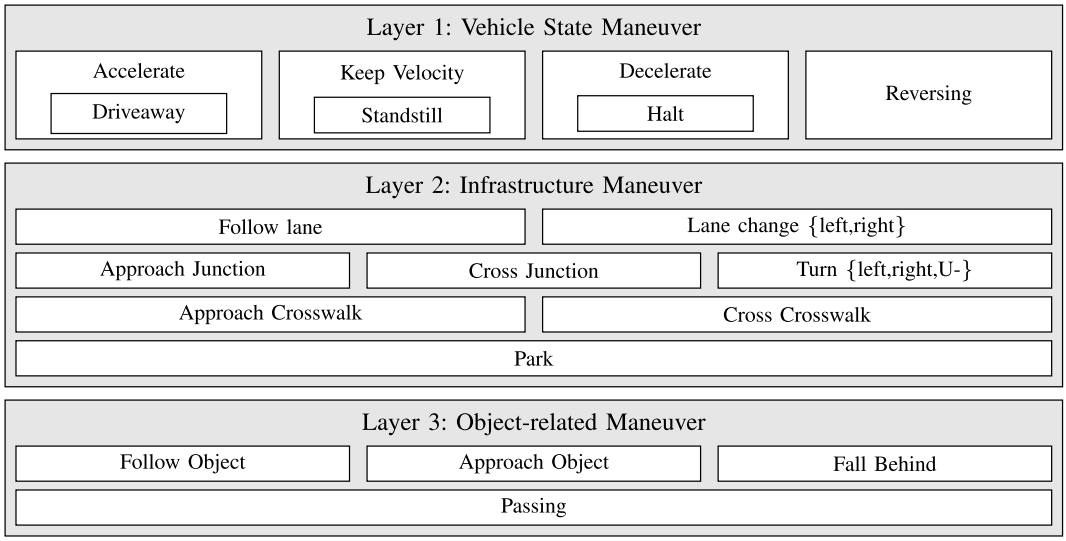


Figure 2.9: The layered maneuver-based model for urban traffic description by Hartjen *et. al.* [61].

2.2.3 Scenario description models

The term "scenario" and other concepts used in this work were introduced in Section 2.1.3. The scenario represents the most abstract concept. As indicated in Figure 2.7, however, there is a wide variety of different scenarios that can occur in the real world. The following section describes relevant studies that present approaches to describing and categorizing scenarios.

Menzel *et al.* present a widely utilized model for defining scenarios, which delineates scenarios across three abstraction levels (see Figure 2.10). The specification of specific properties of scenarios, such as the traffic participants, their positions, and the relationships between them, is conducted with varying granularity.

The most abstract scenario variant according to Menzel *et al.* is the *functional scenario*. Functional scenarios are defined on a semantical level, resembling the textual description of scenarios as used in this work so far, where certain entities in a scenario and their activities are described using natural language. [62] This is illustrated in the left panel of Figure 2.10 which describes a functional scenario with a "*car and a truck [...] driving on a right lane of the road, whereby the car follows the truck*" [62] "*on a two-lane motorway in a curve*"

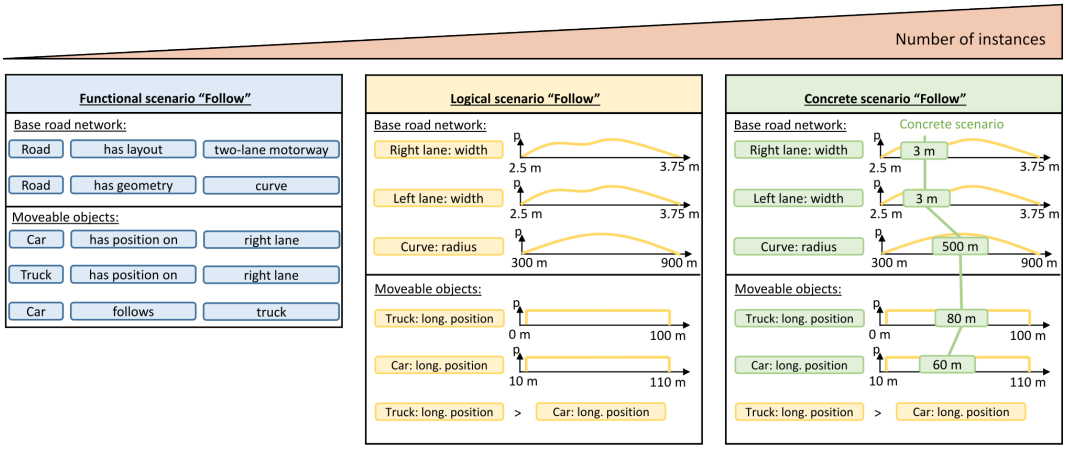


Figure 2.10: The scenario abstraction levels from Menzel *et al.* [62].

[62]. One of the most significant advantages of this abstract level is the utilization of terminology and concepts derived from human language. This facilitates the comprehension and definition of scenarios among different stakeholders.

On the second abstraction level, the semantic terms used to describe the functional scenario are mapped to specific parameters. For each parameter, a value range is defined which may follow a specific distribution, *e.g.*, Gaussian or Uniform. Furthermore, relations between parameters can be defined by conditions as illustrated in Figure 2.10 with the longitudinal position of the truck being greater than the car. This conditional relation is enough to represent the *car follows truck* relation on the functional scenario level. In fact, the logical scenario is a representation of the functional scenario relations to numeric parameters, while the value ranges describe the scope of each parameter in this scenario.

For scenario-based testing (SBT) and especially the extraction of scenarios from real-world traffic data, the *concrete scenario* is of major interest. The concrete scenario is used to describe a realization of the scenario and "*could be used as a basis for test case generation*" [62]. That is, for every parameter that is used to describe the logical scenario, a concrete value is chosen (as illustrated in the right panel of Figure 2.10). Due to the representation of parameters with distributions, established sampling methods can be utilized to create concrete scenario from logical scenario. [62]

This work follows the concepts of scenario abstraction by Menzel *et al.* For the representation of traffic data as scenarios, the scenarios of interest to identify are defined on the semantic level of *functional scenario*. But, instead of defining the parameter values of the relations, those will be derived from real-world traffic data. Hence, the functional scenario definition is employed to extract concrete scenarios which, for instance, can be composed to logical scenarios or provided as tests for SBT.

The approach for scenario description by Menzel *et al.* focuses on defining scenarios, where parameters and attributes are defined with varying granularity across the three different scenario abstraction levels. This allows for deriving concrete scenarios from a rather abstract and semantical scenario description. These concrete scenarios represent

the same abstract scenario but vary due to parameter variations, such as a different turning radius or different velocity of traffic participants. But, as illustrated in the Figure 2.7, there are various scenarios that can occur in the real world. Therefore, for the safety argumentation of AVs, it is necessary to test them across a diverse set of relevant scenarios.

These various scenarios are identified by categorizing the overall scenario space, which is, for instance, defined by the ODD. One such approach is presented by Weber *et al.*, where a semi-formal methodology is proposed to define a scenario catalog for urban areas that is composed of, so-called basic scenarios. The work identifies a variety of basic scenarios and assigns them to high-level scenario categories.

Another approach is proposed by Markkula *et al.* also for the urban area, with a primary focus on interactions between traffic participants within a scenario and the potential conflicts between them. In contrast to the proposal of Weber *et al.*, scenarios are divided into five categories, the so-called space-sharing conflict scenarios. According to Markkula *et al.* [64] a space-sharing conflict is an "observable situation from which it can be reasonably inferred that two or more road users are intending to occupy the same region of space at the same time in the near future". Transferred into the domain of scenarios, a space-sharing conflict scenarios can be categorized as obstructed, merging, crossing, unconstrained and constrained head-on paths as illustrated in Figure 2.11. In the following, those scenarios are discussed briefly.

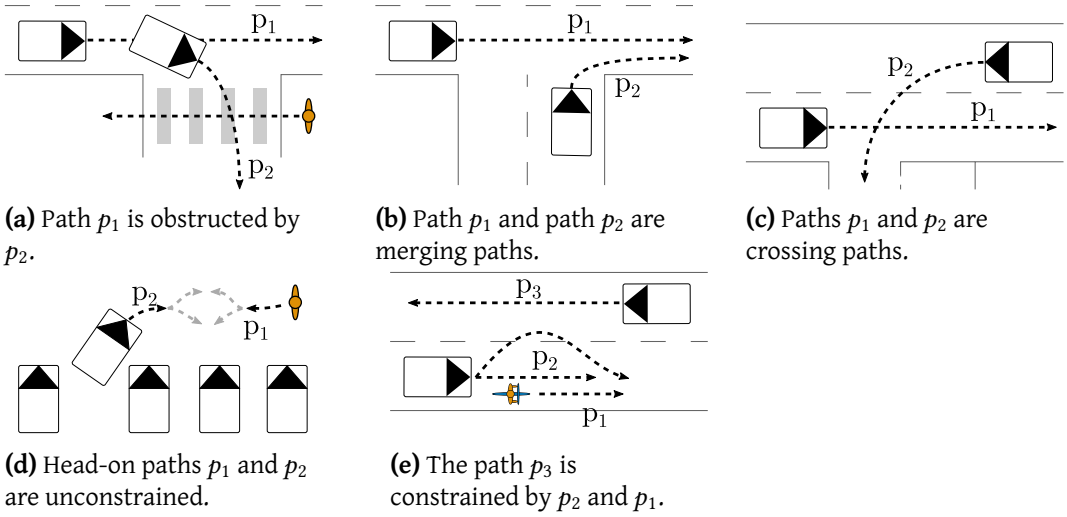
In a space-sharing conflict scenario by an *obstructed path* a traffic participant cannot follow its route since the path is obstructed by another traffic participant. For instance, if a lane on an intersection allows turning right and crossing straight the intersection (see Figure 2.11a), a traffic participant might obstruct the path of another traffic participant that wants to cross the intersection straight, but has to wait until the path becomes free.

If two traffic participants want to leave the intersection in the same direction but have different origins, they have *merging paths*. An example shared-space conflict situation due to merging paths is depicted in Figure 2.11b. The traffic participant with path p_1 wants to cross the intersection straight from west to east, and the traffic participant with path p_2 will perform a right turn leaving the intersection east.

Another variant of a space-sharing conflict scenario is defined by traffic participants with *crossing* routes. An example situation is shown in Figure 2.11c with a traffic participant crossing the intersection straight p_1 and another one performing a left turn p_2. The traffic participant with path p_2 needs to cross the path p_1 to perform the left-turn maneuver. The traffic participants have both different origins and destinations.

The fourth variant is the scenario with *unconstrained head-on paths* where traffic participants move towards each other but can freely adapt their path for collision prevention and to follow their route (indicated by gray paths). Such a situations may occur, for instance, in parking lots where vulnerable road users and other traffic participants move freely as depicted in Figure 2.11d.

The last variant of space-sharing conflict scenario (SSCS) according to Markkula *et al.* [64] is where traffic participants head-on paths might be *constrained*. An example situation is shown in Figure 2.11e where a cyclist p_1 drives on the road in front of a vehicle with path p_2 and another traffic participant p_3 coming from the opposite direction. The head-on paths of the latter traffic participant with path p_3 is constrained in this situation by the



(a) Path p_1 is obstructed by p_2.

(b) Path p_1 and path p_2 are merging paths.

(c) Paths p_1 and p_2 are crossing paths.

(d) Head-on paths p_1 and p_2 are unconstrained.

(e) The path p_3 is constrained by p_2 and p_1.

Figure 2.11: The space-sharing conflict scenarios according to Markkula *et al.* [64].

road user with path p_2 since it could overtake the cyclist with path p_1 and may partly use the oncoming lane.

The benefit of categorizing scenarios according to Markkula *et al.* is its emphasis on the interactions between traffic participants. Particularly in urban environments, scenarios of interest for the safety argumentation of AVs involve situations where the ego vehicle interacts with other traffic participants, experiencing either critical situations or atypical behavioral patterns. Chapter 5 introduces a method for formally defining these SSCS, using the AIM Research Intersection as an example, and identifying potentially interacting traffic participants involved in these scenarios from real-world traffic data. The formal representation of the entities and their relationships has been effectively achieved through the use of knowledge bases based on ontologies. Zipfl *et al.* [65] provide an overview of relevant work in the context of scenario-based testing (SBT). For example, Westhofen *et al.* describe a method for defining critical phenomena using ontologies based on an extended layer model for traffic data description by Scholtes *et al.* [67], which is based on the work of Bagschik *et al.* [59]. They also demonstrate how these phenomena can be identified in real-world data. Since Ontologies has been shown to be an appropriate method for scenario definition, the approach presented in Chapter 5 employs ontologies to formally define SSCS, relevant entities, and the relationships. This is done following a modular approach similar to [66], where different entities and relationships are defined in separate ontologies.

2.3 Digital Road Networks

The description of scenarios not only includes information about the traffic participants involved and the interactions among them. As already illustrated in the layered model Figure 2.8 of Bagschik *et al.* [59] and the scenario abstraction levels by Menzel *et al.* [62] infor-

mation about the roads and traffic infrastructure is required. This information is pivotal for the operation of automated vehicles as a digital representation of the road network helps to understand the vehicle's surrounding. [68] Furthermore, accurate and detailed digital representation of the road and traffic infrastructure helps in the simulation-based testing and validation of theses systems as they enable to create realistic test cases. [69]

There are various formats to represents information about the road and traffic information in so-called *digital maps*, including OpenStreetMap [70], lanelets [71] and the ASAM Standard OpenDRIVE [72]. An extensive overview about this topic is given by Elghazaly *et al.* [73]. In the remainder of this work, the OpenDRIVE format is used to represent road information. A brief description of relevant features of this format is given in the following.

A road network in OpenDRIVE format is, at the top level, described in terms of roads and junctions, while junctions are utilized to connect roads with each other. Furthermore, the road infrastructure is represented using two geometric representations, the *reference lines* and the *lane borders*. All geometric aspects of a road and other properties along the road are defined w.r.t the reference line. This also holds for describing lanes of a road. Their geometric representation is within the reference system of the road, which is defined by the reference line. An example road network in OpenDRIVE format is illustrated in Figure 2.12. It represents the inner-city ring of Braunschweig and the DLR AIM Research Intersection. The reference lines of network's roads are depicted in Figure 2.12a and of the DLR AIM Research Intersection in Figure 2.12b. The roads within the inner intersection area are linked to a junction. This is necessary since the roads leading to and departing from the intersection are linked to the inner intersection's roads via the junction. This property is utilized in Chapter 5 for scenario definition. The OpenDRIVE format is designed to represent the traffic infrastructure with high accuracy. For this purpose, lane borders are utilized to represent the shape of roads or other areas, such as parking lots, traffic islands or sidewalks. The representation of the DLR AIM Research Intersection on lane-level is illustrated in Figure 2.12c clearly showing the shape of the individual lanes and sidewalks.

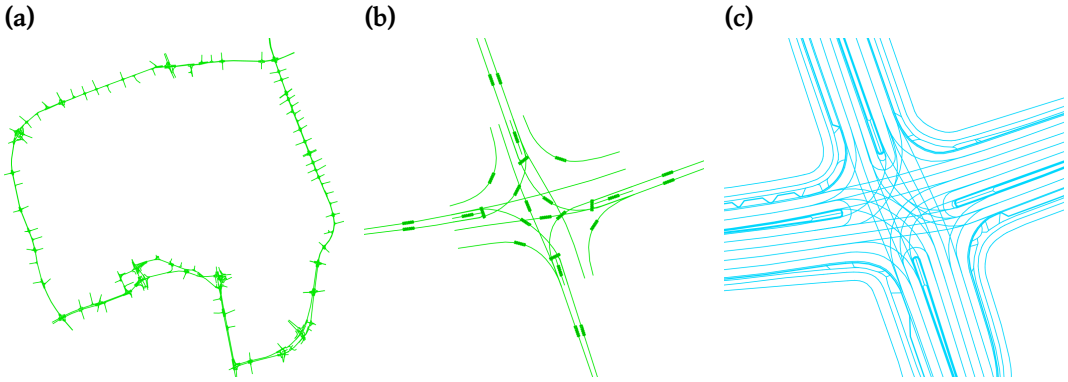


Figure 2.12: A digital representation of the inner-city ring of Braunschweig (2.12a) in OpenDRIVE format ([74]). The reference lines of the DLR AIM research intersection are illustrated in 2.12b and the lane borders in 2.12c.

2.4 Surrogate measures of safety

Scenarios are typically defined using three abstraction levels as presented in previous section and are categorized according to the interaction among the traffic participants involved. The logical scenario describes the variance of a functional scenario and thus limiting the overall scope of concrete scenario instances. But, based on a functional or logical scenario, any number of concrete scenarios can be derived. For scenario-based testing (SBT), however, not every concrete scenario is relevant. For instance, complex scenarios, those with rare phenomena, or scenarios with critical interactions between traffic participants as illustrated at the top of the safety pyramid by Hydén *et al.* illustrated in Figure 1.1 by may be worth testing more extensively on special proving grounds or in simulations.

For the objective evaluation of a scenario in terms of traffic safety, Surrogate Measure of Safety (SMoS) are widely adopted. Various metrics exist to quantify traffic safety based on interactions between road users. While an in-depth overview of this topic is beyond the scope of this work, refer to Wang *et al.* [75] for an extensive review. In the following, three basic time-based SMoS are revisited which are used throughout the work.

2.4.1 Time to collision

The time to collision (TTC) is a SMoS proposed in 1972 by Hayward [76] and used to evaluate the risk of rear-end collisions in traffic scenarios. The TTC represents the "*time required for two vehicles to collide of they continue at their present speeds and on the same path*" [76]. Let $v_1(t), v_2(t)$ be the velocity of both participants, the $TTC(t)$ at time t is estimated as

$$TTC(t) = \frac{\mathbf{p}_2(t) - \mathbf{p}_1(t)}{v_2(t) - v_1(t)} \tag{2.6}$$

with $\mathbf{p}_2 - \mathbf{p}_1$ the distance between both road users positions at time t. An advantage of the TTC is that it allows to quantify the collision risk between traffic participants continuously and not only for discrete points. However, the measure does not account for driving behavior adaptations or vehicle dynamics, such as steering or breaking.

2.4.2 Gap time

Another SMoS that is worth mentioning is the Gap Time (GT). That measure is used in different contexts, including the safety assessment of situations [77] and behavioral analysis in left-turn scenarios with pedestrian-vehicle interactions [78], or for maneuver planning [79] with vehicle-vehicle interactions. In the first two cases, the GT denotes the difference in time of arrival of both participants reaching the conflict point as illustrated in Figure 2.13, and in the latter case, the GT denotes the time gap between two successive vehicles. If we assume that $d_1(t), d_2(t)$ are the distances to the conflict point along the paths of both road users at time t, the GT is defined as

$$GT(t) = \frac{d_2}{v_2(t)} - \frac{d_1}{v_1(t)} \tag{2.7}$$

with $v_1(t), v_2(t)$ the velocity of both road users at time t. Note that for the case of following vehicles, the conflict point is typically at the rear end of the leading vehicle. Thus, the right term of (2.7) becomes zero and the GT in this case is the arrival time of the following vehicle. But, in both cases, a smaller GT indicates a more critical situation. The Gap Time is utilized in an application in Section 6.4.3, which analyzes the merging behavior of traffic participants on the highway.

2.4.3 Post encroachment time

Another time-based SMoS that is used to measure the time proximity to a crash between two traffic participants is the PET. It was proposed in 1987 by Allen *et al.* [80] as a measure to indicate the severity of a conflict especially at intersections and other conflict zones. Figure 2.13 illustrates the PET at an intersection with a road user turning left (p_1) and another one driving straight (p_2). The PET is defined as

$$PET = t_2 - t_1 \tag{2.8}$$

and the time difference between the moment p_1 leaves the conflict point t_1 and the moment p_2 arrives at the conflict point t_2. The sign of the PET value indicates the order of the road users entering/leaving the conflict zone. In general, a smaller PET value indicates a more severe conflict situation, while a PET value of zero denotes a crash. Other than the TTC, the PET measure is not estimated continuously, but is evaluated once for two road users. The PET measure is used throughout this work not only to assess scenarios in terms of traffic safety in Section 6.4.3. The fact that the PET can be estimated for two traffic participants only if they share the same spatial location is utilized as an indicator in Chapter 5 to identify crossing scenarios from real-world data. Moreover the sign of the PET and the relationship between a traffic participant merging on the highways to other traffic participants on the highway is combined in Section 6.4.3 to categorize on-ramp scenarios on a highway.

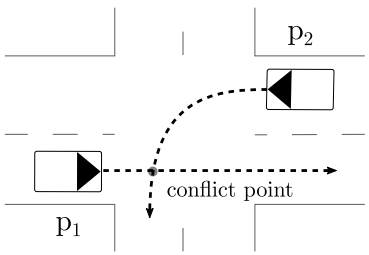


Figure 2.13: Example scenario at an intersection with a road user p_1 turning left and another road user p_2 driving straight.

2.5 Application-specific foundations

As mentioned in the beginning of this chapter, relevant terms and concepts are introduced thus far that are relevant for the remainder of this work. For the sake of clarity, foundations that are relevant only for specific application are introduced in the corresponding chapters. For the sake of completeness, an overview is given in the following.

In Section 3.1, the concept of a primitive is described in more detail and is related to the traffic descriptions models illustrated in Section 2.2 using examples. Furthermore, a brief review of the Hidden Markov Model (HMM) and Dynamic Time Warping (DTW) is given, since both methods are utilized for primitive-based representation of traffic data and the identification of maneuvers based on it.

In Section 5.1, terms and concepts are introduced for knowledge representation since this is the foundation of the proposed approach for systematic scenario identification. In particular, a short introduction about Knowledge Base (KB), ontologies and SPARQL Protocol And RDF Query Language (SPARQL) is given.

2.6 Summary

This chapter introduces fundamental concepts for the scenario-based representation of traffic data. it defines key terms such as position, pose, and trajectory, which describe the spatiotemporal representation of traffic participants within traffic data. Moreover, emphasis is placed on the hierarchical structure of scenarios, which includes scenes, primitives, maneuvers, and scenarios. Primitives are the building blocks to semantically represent traffic data, capturing basic actions and interactions among entities. Furthermore, three distinct variants of a scenario description are presented, which differ in their level of abstraction from functional, over logical to concrete scenarios. From real-world traffic data, concrete scenarios are typically extracted and are utilized for scenario-based testing (SBT). However, for the latter, scenarios are typically filtered according to certain criteria such as traffic safety, rareness or complexity. The section also revisits three SMoS for the quantitative analysis of traffic safety in scenarios that are used throughout the work.

Chapter 3

Representation of traffic data with primitives

This work motivates the representation of traffic data on a semantical level by breaking down the description of the real world traffic data to a set of unique non-overlapping attributes: the so-called primitives. In this section, a framework is presented that allows to systematically represent traffic data using primitives and the identification of relevant maneuvers given a use case or scenario definition addressing O.III. A key to this is the definition of a primitive domain, the representation of traffic data in these, and the definition of maneuvers on a semantical level using the primitives as defined in O.IV. Two case-studies demonstrate the presented approach facing two different use cases. In the first case study lane-change maneuvers will be identified and extracted from a traffic dataset collected on a German highway with a VuT. The second case study demonstrates the application of the approach for the problem of route estimation on the urban DLR AIM Research Intersection.

3.1 Foundation

As mentioned in Chapter 2, application-specific foundations are described in the relevant chapters. In the following, an overview of primitives and the relation to existing concepts is given. Moreover, concepts and methods are briefly introduces that are used throughout this chapter.

3.1.1 Overview

Following the traffic description of Bagschik *et al.* and Hartjen *et al.* from Section 2.2 and putting the domain of primitives into context, this work focusses on the primitives related to the levels *Road-level* and *Objects* — but from the perspective of the *ego vehicle* as in [43]. That is, the focus is on the object of main concern in a specific situation. For instance, if the use case is to analyze the left turn behavior on an intersection, the ego vehicle is the traffic participant performing that left turn. If the use case is, however, to identify situations where a vehicle crosses an intersection straight has to decelerate to mitigate a collision with a left turning vehicle, the ego vehicle is the one driving straight ahead. Hence, the perspective is always dependent on the use case, the scenario of interest or the research question. In fact, the focus is on a subset of maneuvers defined by Hartjen *et*

al. for demonstration purpose and to illustrate a primitive based approach for maneuver description.

As described in Section 2.1.3, a primitive can describe a certain state of an entity. The domain of that state space is dependent on the context. For the *Road Level*, which might be used to identify, *e. g.*, lane-change maneuvers, the context could be the position relative to a lane of the digital map, that contains the geometric information of the roads, if we take the distance to road lanes from the ego perspective into account, *i. e.*, the representation of the trajectory in the road reference coordinate system (see Section 2.1.2). The domain could be, in this case, the space of relative positions. But, a primitive can also denote how the ego object moves towards a lane and thus describes the *dynamic behavior*. In that case, the domain is the change of the relative position and the context the relative movement to a road lane.

That methodology can also be applied to the *Objects* level, to either describe the relationship between the ego vehicle and other traffic participants, or the ego vehicle's state and movement. An exemplarily representation of the vehicle's movement is illustrated in Figure 3.1. The movement is generally described by four categories. The category *driving direction* denotes if the traffic participant is moving forward, backward or not at all using three primitives. The second category *velocity* describes how fast the vehicle moves and the other two categories the traffic participant's *acceleration* and *deceleration* — all three categories with five primitives. This is effectively a different view on the traffic participant's state as defined by Hartjen *et al.* in Layer 1 (see Figure 2.9). The traffic participant's movement is fully described using four categories and include a semantical representation of the actual magnitude, e.g., *fast velocity* or *strong acceleration*.

Driving direction	**Velocity**	**Acceleration**	**Deceleration**
• Backward	• Standstill	• None	• None
• None	• Slow	• Low	• Low
• Forward	• Moderate	• Moderate	• Moderate
	• Fast	• Strong	• Strong
	• Very fast	• Very strong	• Very strong

Figure 3.1: The movement representation of traffic participants based on primitives.

Another example for a primitive-based description of traffic data is illustrated in Figure 3.2. It shows the positional relationship between road users if they would be represented in the reference frame defined by the road (see Section 2.1.2). In general, the relationship is represented in the longitudinal and lateral direction. In the first direction, the object is either *Behind*, *Next to* or *In front* of another object. The second direction describes the lateral position with three primitives. The third category illustrated in Figure 3.2 denotes an exemplarily representation of the positional change in the longitudinal direction between two objects with three states.

Longitudinal	Lateral position	Longitudinal change
• In front	• Left	• Follow
• Next to	• Right	• Approach
• Behind	• Level with	• Leave

Figure 3.2: The positional relationship between two traffic participants.

This representation of the positional relationship can be useful for the use case of, e.g., identifying overtaking scenarios. An example of such an overtaking scenario is illustrated in Figure 3.3 with the ego vehicle (in blue) overtaking the challenger vehicle (in green). Initially, the ego vehicle is in the same lane as the challenger, but reduces the distance to it. The ego vehicle then changes lanes to the left and passes the challenger, finally changing back to the right lane and increasing the distance to the challenger.

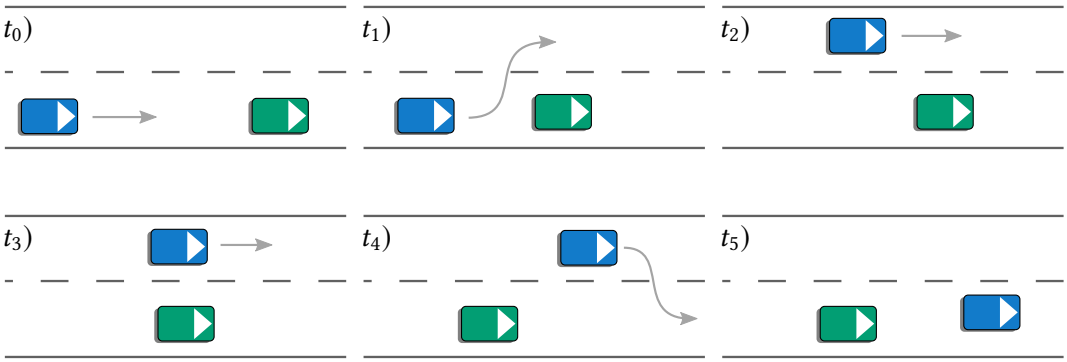


Figure 3.3: An example overtaking scenario with the ego vehicle in blue and the challenger in green.

Instead of this textual, or functional, description of the scenario evolution, it can be represented using the positional relationship primitives from Figure 3.2 for the six scenes illustrated in Figure 3.3. In Table 3.1 every row represents the scenario evolution w.r.t a positional relationship category. The first row focus on the *longitudinal* relationship and clearly shows that the ego vehicle is initially behind the vehicle and finally in front of it indicating that it overtakes the challenger. The second row describes the *lateral* relationship and is required to properly describe this scenario, since it shows that the ego vehicle is at first in the same lane as the challenger, then left of it and finally back in the same lane. Without the lateral position information, the ego vehicle could, for instance, be on the left lane over the full course of the scenario, thus actually describing another variant of an overtaking scenario. The third row represents the longitudinal positional change of the ego vehicle to the challenger. Although the category does not seem to be mandatory for the description of the scenario, it is indeed necessary. Imagine a scenario where the left lane is occupied by another vehicle and the ego vehicle must wait behind the chal-

lenger before changing lanes. Both scenarios can be described by the same course w.r.t the longitudinal and lateral position. The information about the positional change allows a distinction to be made between both scenarios. Because in the case of the scenario with occupied left lane, the ego vehicle would *Follow* the challenger on the same lane until the other lane is clear as opposed to the scenario in Figure 3.3.

Table 3.1: The positional relationship of the ego vehicle to the challenger over the course of the overtaking scenario illustrated in Figure 3.3.

Relationship \ Scene	t_1	t_2	t_3	t_4	t_5
Longitudinal	Behind	Behind	Next to	In front	In front
Lateral	Level with	Left	Left	Left	Level with
Longitudinal change	Approach	Approach	Approach	Leave	Leave

The primitive-based representation of traffic data not only allows to describe it on a semantical level. It also enables the the description and identification of maneuvers based on the evolution of primitives. For instance, in the textual description above, it is stated that the ego vehicles *passes* the challenger, which could be viewed as a specific maneuver that the ego vehicle performs. In fact, Hartjen *et al.* [61] defined *Passing* as a maneuver to represent the relationship between two objects (see Figure 2.9). With the primitives illustrated in Figure 3.2, this maneuver can be decomposed for the sake of maneuver identification. Using the longitudinal positional relationship, that maneuver can be represented as the sequence (Behind, Next to, In front).

Another maneuver that is used to describe the scenario is *changing lanes* with the directional information (*left* and *right*). In the layered maneuver-based model of Hartjen *et al.* this is a infrastructure maneuver, since it describes the relationship between an object and the digital map (infrastructure). [61] This maneuver can also be represented using the primitives defined in Figure 3.2. The lane change to the left can, for instance, be represented with the sequence (Level with, Left) and to the right with (Left, Level with) using the lateral positional primitive. In the first case study (see Section 3.3), another representation will be proposed that utilizes primitives describing the relationship between the object and the infrastructure instead of the inter-object relationship.

For both examples, it is possible to represent the traffic data using the primitives and define the maneuvers as a sequence of the primitives. For the use case of maneuver identification in real-world driving data, we need to find those patterns and extract them.[1]

3.1.2 Hidden Markov Model

For the representation of real-world traffic data as primitives, various methods exist that can be utilized. In this chapter, it will be demonstrated that a Hidden Markov Model (HMM)

[1] It is worth noting that maneuver identification is one use case. Another use case is searching specific situations. The primitive-based representation allows searching on a semantical level, e.g., by a road user's velocity or acceleration using the primitives defined in Figure 3.2.

is one of these approaches. While this method originates from the domain of speech recognition is was successfully applied to solve various real-world problems. [81, p.43][82] An HMM is a probabilistic method that allows to infer the "hidden" states of a process giving the information this process emits. In particular, given a sequence of observations of a system, an HMM can be utilized to predict the most probable sequence of states. [82]

A HMM λ is defined as $\lambda = (\boldsymbol{A}, \boldsymbol{B}, \boldsymbol{\pi})$ where $\boldsymbol{A}$ is the transition probabilities between the hidden states, $\boldsymbol{B}$ the observation probability distribution conditioned on the current hidden state, and $\boldsymbol{\pi}$ the initial state probabilities. [83]

With the representation of traffic data in terms of primitives, we may transform continuous variables to discrete ones with a finite value range. The longitudinal relationship between two traffic participants, for instance, may be represented by three primitives as illustrated in Table 3.1, which could be derived from the actual distance given in meters. For this purpose, a Gaussian Mixture Model (GMM) is typically utilized to model the observation probability distribution, which is defined by the means and covariances of the mixture components. [83]

For the representation of traffic data as primitives, two problems must be solved. First, the parameters of the HMM needs to be defined. Second, given the sequence of observations, such as distances between traffic participants, and a HMM, the sequence of stated, or longitudinal primitives in the example above, need to be estimated. According to Bilmes [84], these are typical problems for which solutions are available. The parameters of the HMM can be estimated from a sequence of observations using the Baum-Welch algorithm. Furthermore, given the parameters and the sequences of observations, such as the distances between traffic participants, the sequence of states can be estimated using Viterbi algorithm. [84]

3.1.3 Dynamic Time Warping

The first case study of this chapter will demonstrate how to identify maneuvers using the primitive-based representation of traffic data. While the HMM is employed for this representation, maneuvers still need to be identified in this sequence of primitives. This chapter will demonstrate that the DTW can be utilized for this purpose.

DTW originates from the domain of knowledge discovery and data mining and was proposed by Berndt *et al.* [85] to identify patterns in time-series data. The method uses dynamic programming to compare a time-series with a specific template, while aligning the time domain. This allows to find patterns in the time-series with a wide variation in timing, because it is "both ignoring global and local shifts in the time dimension" [86].

DTW creates a warping path $\mathcal{W} = w_1, w_2, \ldots, w_N$ from two time-series $\mathcal{A}, \mathcal{B}$ with a $|A| \times |B|$ matrix. Each cell in this matrix contains the distance between elements of $\mathcal{A}$ and $\mathcal{B}$. The warping path $\mathcal{W}$ is the one minimizing the cumulative distance over potential paths defined as

$$\mathrm{DTW}(\mathcal{A}, \mathcal{B}) = \min_{\mathcal{W}} \sum_{i=1}^{N} w_i \tag{3.1}$$

and that represents the "distance" between both time-series, which allows to differentiate of various templates. [85] This property is used in the case study in Section 3.3 to classify extracted partitions of a driving sequences with a lane-change maneuver variant.

3.2 Framework for primitive based maneuver identification

In the previous section, the concept of a primitive is described in more detail while the examples show that various phenomena and relations among entities can be defined using primitives. In the following, a framework is introduced that serve as a guideline to represent real-world traffic data using primitives.

Let us recap that $\mathcal{X}$ defined in (2.3) is a time series of state-space vectors that represents the trajectory of a traffic participant, including information about the environment, that is defined as $\mathcal{X} = (\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_T)$, with T the trajectory's length and each $\mathbf{x}_t \in \mathcal{X}$ is a (partial) scene representation at time t.

Furthermore, let $\mathcal{K} = \{\mathbf{k}_1, \mathbf{k}_2, \ldots, \mathbf{k}_N\}$ be a set that represents the N contexts and $\mathcal{D} = \{\mathbb{D}_1, \mathbb{D}_2, \ldots, \mathbb{D}_N\}$ a set representing the N primitive domains, where each context allows to represent a scene in the corresponding domain of primitives. For that purpose, let $t(\mathbf{k}) : \mathbf{x} \rightarrow y$ be a transformation to associate a scene $\mathbf{x}$ to an entity $y \in \mathbb{D}$ of a primitive domain $\mathbb{D}$ using the context $\mathbf{k}$. That is, a scene can only be associated to a single primitive of a primitive domain. A primitive domain is, for instance, the lateral position relationship in Figure 3.2 and the context the distance to the lane marking, since it allows to infer the lateral position. In fact, this information will be used in the first case study in Section 3.3.

Moreover, let $\mathcal{Y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_T)$ be a time series with the same length as the trajectory $\mathcal{X}$. Every element $\mathbf{y} \in \mathcal{Y}$ is a vector denoted as $\mathbf{y} = \left[y_1, y_2, \ldots, y_{|\mathcal{D}|}\right]^T$ which dimension is determined by the number of primitive domains. That is, the vector $\mathbf{y}_t$ is the representation of the scene $\mathbf{x}_t$ in the domains of primitives at time t. For the overtaking scenario above, $\mathbf{y}$ would be a three-dimensional vector since three categories or domains (see Table 3.1) are used for the description. Hence, we follow a different approach than Wang *et al.* [45] by allowing a pose of a traffic participant to be represented with multiple primitives, which was also illustrated in Figure 2.6.

In addition, let $\mathcal{M} = \{\mathcal{M}_1, \mathcal{M}_2, \ldots\}$ represent the maneuvers that should be extracted from a trajectory. Every element $\mathcal{M}_\mathbb{D} \in \mathcal{M}$ denotes the maneuvers that can be represented using the primitive domain $\mathbb{D} \in \mathcal{D}$. Every member $\mathbf{m} \in \mathcal{M}_\mathbb{D}$ is a sequence denoted as $\mathbf{m} = (m_1, m_2, \ldots)$ with $m \in \mathbb{D}$ defining the maneuver *signature*. That is, the maneuver signature allows to describe the course of the maneuver in the primitive domain.

Based on these definitions, for a certain use case the time-series $\mathcal{X}$ can be represented as primitives $\mathcal{Y}$ and maneuvers $\mathcal{M}$ can be derived based on them by defining the following entities — not necessarily in the given order.

- **Maneuver**: Define the relevant maneuvers $\mathcal{M}_k = \mathcal{M}_1, \mathcal{M}_2, \ldots$ for every context $k \in \mathbf{k}$ according to a use case. Every element should represent the maneuver signature in the domain of primitives.

- **Domain**: For every context $\mathbf{k} \in \mathcal{K}$, specify the domain $\mathbb{D}_\mathbf{k}$ that defines the primitives. Since primitives represent certain aspects or states of the maneuver, the domain is typically a finite set of real-valued numbers.
- **Context**: Define the primitive contexts $\mathcal{K}$ that are relevant for the use case and that allow representing the trajectory $\mathcal{X}$ in the domains $\mathcal{D}$ of primitives.
- **Transformation**: For every pair $(\mathbf{k}, \mathbb{D})$ of a context $\mathbf{k}$ and its primitive domain $\mathbb{D}$ define a transformation $t(\mathbf{k})$ that refers a state-space vector $\mathbf{x}$ to an entity of the domain $\mathbb{D}$ according to the context $\mathbf{k}$. That is, $t(\mathbf{k})$ will transform the driving data $\mathcal{X}$ into the domain $\mathbb{D}$ of primitives $\mathcal{Y}$ via the context $\mathbf{k}$.
- **Maneuver classification**: For every context $\mathbf{k} \in \mathcal{K}$ that has multiple maneuver definitions, define an function that unambiguously associates a time series in the domain of primitives $\mathcal{Y}$ to a maneuver $\mathbf{m} \in \mathcal{M}_\mathbf{k}$ based on the maneuver signatures.

To demonstrate the consistency and application of that framework and to propose methods to solve the formulated issues, two case-studies are presented in the following. In the first case study, lane-change maneuvers should be identified and extracted from real-world drivings on a german highway that was collected during the FASva project[2]. In the second study, the framework is utilized to solve the problem of route finding on an urban intersection using real-world drivings from the DLR AIM Research Intersection.

3.3 Case study: Identification and extraction of lane-change maneuvers

The following study will demonstrate the approach of representing real-world driving data with primitives for the use case of lane-change identification and extraction[3]. The aim is to identify different variants of lane-change maneuvers solely based on different combinations of primitives.

3.3.1 Primitive domain

The domain of primitives for lane-change maneuvers is defined by dividing the maneuver into non-overlapping sub states so that the association to any of the primitives is distinct. This domain is derived from Figure 3.4 that depicts how a lane-change is decomposed into sub states or primitives in this work. Note that the division must not be too granular to ensure that the states can be represented unambiguously. But also not too coarse so that relevant information might be hidden in a state. The decomposition depicted in Figure 3.4 allows to define different lane-change maneuvers. Furthermore it enables to analyze driver behavior in certain sub states in identified lane-changes and thus might also serve as a representation for other use cases.

[2] https://www.hs-emden-leer.de/studierende/fachbereiche/technik/projekte/fasva
[3] This approach and application for lane-change identification was published in [87]

Each primitive depicted in Figure 3.4 represents the location of the object on the lane. If the object is in the state `Idle`, the vehicle's center is in the middle of the lane. If the object is moving towards a lane marking (left or right) it will eventually change to the state `Approach`. The state `Cross` denotes that the object is crossing the lane marking and `Change` that it is changing towards the destination lane. If the object reaches the state `Depart` than, as depicted in Figure 3.4, the majority of the object is in the destination lane. Eventually, that object keeps moving towards the destination lane so that it enters that lane (in state `Enter`) and maybe reaching the `Settle` state. This state denotes that the object's full body is on the destination lane. If the object is on the center of the destination lane, it will be again in the state `Idle`.

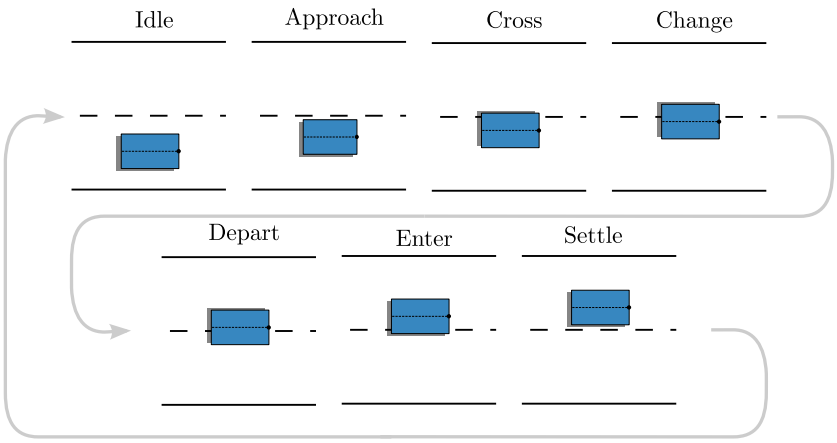


Figure 3.4: A lane-change is divided into six sub states representing the primitives.

Remark that the association to a primitive should be distinct in every case. This is, however, not the case for the primitives shown in Figure 3.4. The distinction between `Approach` and `Settle`, but also `Cross` and `Enter` as well as `Change` and `Depart` might be difficult. If we should infer the primitive solely based on the current view on the situation (the scene) it is hard to state whether an object is either in the state `Approach` and thus maybe starting a lane-change or already in `Enter` and thus finishing a lane-change. That differentiation can be made based on the past movement, but this leads to the problem of defining the length of the history to correctly resolve the state for all possible situations.

Instead of keeping track of the past movement to resolve the current primitive, the domain of primitives is defined by changing the perspective on Figure 3.4 to a lane-orientation point of view (see Figure 3.5). That is, the domain of primitives only represents the states of a lane-change maneuver that relates the vehicle to a lane. This allows to drop the primitives `Depart`, `Enter` and `Settle` since in those three states the majority of the vehicle is on the destination lane. For instance, if an object is, in the state `Enter` in Figure 3.4, it will be in the state `Approach` in the view of the destination lane. Since the states `Approach`, `Cross` and `Change` can be decomposed according to the relative position of the object on the lane (left or right), the domain of primitives $\mathbb{D}$ is defined as

$$\mathcal{D} = \left\{\mathbb{D}\right\} = \left\{\left\{\texttt{Idle}, \texttt{ApproachLeft}, \texttt{ApproachRight}, \texttt{CrossLeft}, \texttt{CrossRight}, \texttt{ChangeLeft}, \texttt{ChangeRight}\right\}\right\} \tag{3.2}$$

and depicted in Figure 3.5. For all primitives except *Idle* the primitive also contains the information about the relative position on the lane.

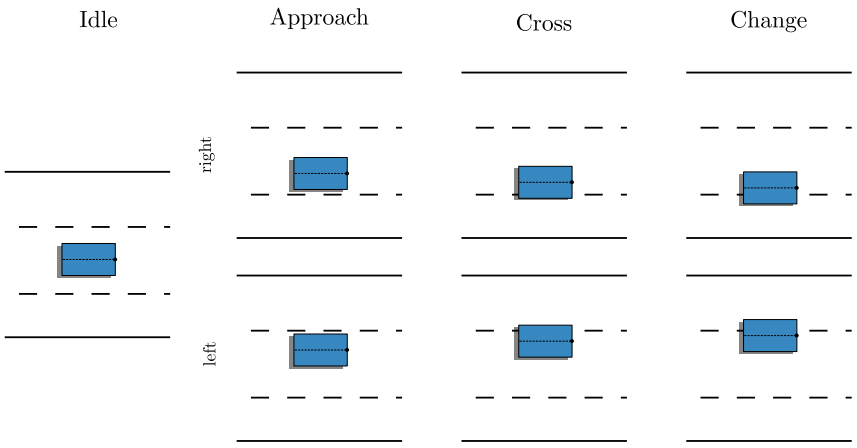


Figure 3.5: The sub states of a lane-change maneuver used as domain of primitives.

3.3.2 Primitive context

Although the context of the previously defined domain of primitives in (3.2) was not explicitly mentioned yet, the proposed lane-orientation point of view implicitly defines it. That is, the context is the lateral distance to road markings since this is decomposed into the defined primitives. Note that the vehicle dimension must be known, too. The definition of this context has the benefit that the proposed approach can be employed to traffic data independent of the data source. That is, as long as the lateral distances and the vehicle dimension is known, the information could be collected by a vehicle with appropriate sensors, traffic infrastructure and even drones.

To support situations with arbitrary road and vehicle widths, a transformation is applied. If the distances to the next left and right lane marking are provided by a vehicle with appropriate sensors, a lane-change to the next left lane could be represented as in Figure 3.6. In that figure, the vehicle's width and the distances are depicted to the next left d_c^l and next right d_c^r lane from the center of the vehicle. This time-series represents a left lane-change since the distance to the next left lane d_c^l tends towards zero and jumps to a distance of approximately 3.5 meter. The distance to the next right lane d_c^r moves in the same direction but jumps from approximately -3.7 to zero meter. The reason for this behavior is that if the vehicle's center crosses the left lane marking, than the vehicle is assumed to be on that lane. Due to that, the signal d_c^l will denote the distance to the left marking of that new lane and d_c^r to the right marking that was the left one before.

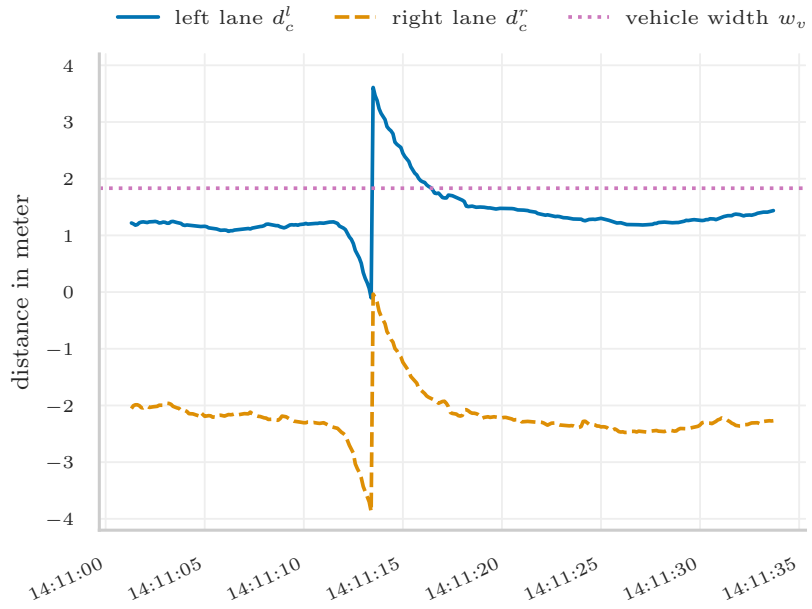


Figure 3.6: An example lane-change to the next left lane represented with the chosen features.

This characteristic also defines the value range for both signals — they are limited by the width of the lane. In fact, for the distance to the next left lane from the vehicle's center d_c^l, the value range is defined as $[0, w_l]$, whereas the distance to the next right lane is $[-w_l, 0]$ with w_l denoting the width of the lane. Furthermore, since both signals are measured from the center of the vehicle, they allow to infer the width of the current lane w_l with

$$w_l = -d_c^r + d_c^l \tag{3.3}$$

since $d_c^r \leq 0$ and $d_c^l \geq 0$. In fact, the constant offset between both signals visible in Figure 3.6 is the width of the current lane.

Instead of using the lateral distances to the next road lane markings, the distance from the vehicle's center to the current lane's center d_c^c is utilized. The motivation of this approach is also founded in Figure 3.5. Since for all states in the depicted domain, the vehicle is in the center lane and the actual primitive can be inferred from the lateral distance of the vehicle to the lane's center. That is, if the vehicle is in the center of the lane and thus in the primitive *Idle*, the lateral distance d_c^c to the lane's center is close to zero. If the vehicle moves towards the left lane, the distance d_c^c would also increase until the vehicle's center crosses the lane marking.

A schematic overview of the available features and the derived ones is depicted in Figure 3.7. Information that is assumed to be known is denoted as bold letters, whereas all other quantities are derived. For now, we focus on the distance to the current lane's cen-

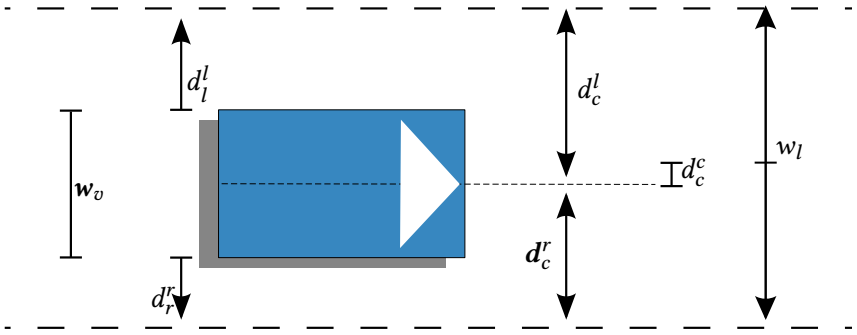


Figure 3.7: An overview of the available features from the context (in bold) and all other features that are estimated based on them for primitive modeling.

ter from the vehicle's d_c^c denoted as the black dot on the front of the vehicle. Based on the definition of the lane width in (3.3) the distance d_c^c is defined as

$$d_c^c = \frac{w_l}{2} - d_c^l \tag{3.4}$$

+

so that $d_c^c \in [-\frac{w_l}{2}, \frac{w_l}{2}]$. Furthermore, $d_c^c \to -\frac{w_l}{2}$ if the vehicle moves toward the right lane and $d_c^c \to +\frac{w_l}{2}$ if it moves towards the left lane, following a right-handed coordinate system with the abscissa being parallel to the lane's heading. Note that (3.4) can also be defined using d_c^r by

$$d_c^c = \frac{w_l}{2} + d_c^r \tag{3.5}$$

if the lane width is known. In fact, due to the relation defined in (3.3), the distance d_c^c can be estimated as long as at least two values are known. That is, the distance d_c^c can also be inferred directly using only d_c^l and d_c^r by inserting (3.3) into, for instance, (3.4) so that

$$\begin{aligned} d_c^c &= \frac{w_l}{2} - d_c^l \\ &= \frac{-d_c^r + d_c^l}{2} - d_c^l \\ &= -\frac{1}{2}\left(d_c^r + d_c^l\right) \end{aligned} \tag{3.6}$$

and thus provides a certain degree of flexibility. That is, the appropriate equation needs to be chosen according to the information that is available. If the information is provided by a vehicle, information might be missing at certain point in times. Especially if cameras are used to locate the lane markings, occlusion in the field of view might lead to missing

information. In that case, the most likely value could be chosen according to the available information. But this is out-of-scope of this work.

Although we have now derived a distance metric that represents the position of a vehicle in a lane as depicted in Figure 3.5, the problem of changing lane widths still exists. However, since the lane width is already defined in (3.3), this can be solved straightforward by normalizing the distance d_c^c according to the lane width so that (3.4) becomes

$$\begin{aligned} d_c^c &= \frac{1}{\frac{w_l}{2}} \left(\frac{w_l}{2} - d_c^l \right) \\ &= 1 - \frac{2d_c^l}{w_l} \end{aligned} \tag{3.7}$$

which is evident from Figure 3.4 because $\frac{d_c^l}{w_l} \to 1$ if the vehicle moves to the right border and $\frac{d_c^l}{w_l} \to 0$ if it moves to the left so that $d_c^c \in (-1, 1)$.

The result after transforming the time-series from Figure 3.4 showing the lane-change to the left using (3.7) is depicted in Figure 3.8. The figure shows the estimated lane width w_l and the normalized distance d_c^c. The jump in the value range is still present but the time-series is horizontally flipped compared to Figure 3.4 since a positive value indicates that the vehicle is on the left side of the lane and a negative on the right side.

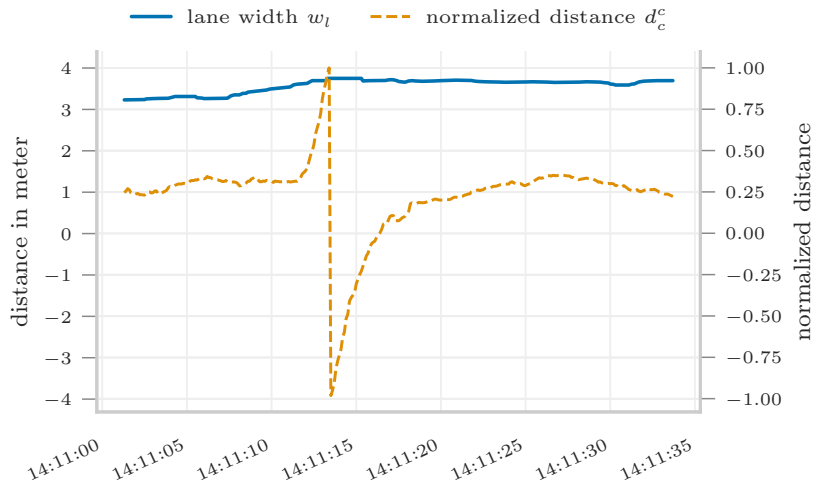


Figure 3.8: The example lane-change from Figure 3.4 represented with the normalized distance denoting the location of the vehicle in the lane independent of the lane width.

The normalized distance now allows to infer the location of the vehicle in the lane. We introduce another feature that states whether the vehicle is in the lane or cross any lane borders to ensure a distinct differentiation. For that purpose, the distances from the vehicle's left side to the left lane d_l^l and from the vehicle's right side to the right lane marking d_r^r are used (see Figure 3.7). The aim is to divide the domain into two subdomains with the first one representing the vehicle driving within a lane and the second one between two

lanes. If we assume that the vehicle width w_v is known the distance to the left marking is defined as

$$d_l^l = d_c^l - \frac{w_v}{2} \tag{3.8}$$

and the distance to the right marking as

$$d_r^r = d_c^r + \frac{w_v}{2} \tag{3.9}$$

from the left and right side of the vehicle respectively. They are now employed to indicate that the vehicle crosses any lane marking $c \in \{-1, 0, 1\}$ by

$$\kappa = \begin{cases} -1 & \text{if} \quad d_l^l < 0 \\ 1 & \text{if} \quad d_r^r > 0 \\ 0 & \text{else} \end{cases} \tag{3.10}$$

to denote that the vehicle crosses the left $\kappa = -1$ or the right $\kappa = 1$ marking, or is within the lane $\kappa = 0$. Hence, the contexts for the use case of lane-change identification is defined as

$$\mathcal{K} = \left\{ \mathbf{k} \right\} = \left\{ \begin{bmatrix} d_c^c \\ \kappa \end{bmatrix} \right\} \tag{3.11}$$

using the normalized distance d_c^c from the vehicle's center to the lane center and a κ to indicate the relative position of the vehicle w.r.t the lane.

3.3.3 Transformation of road user trajectory

For the representation of driving data in terms of primitives for the use case of lane-change identification, a transformation function needs to be defined. This could, for instance, be a rule-based method that annotates a sample with a primitive based on a set of thresholds defining the borders between them. Such an approach is, however, sensitive to outliers in the data and needs to be fine-tuned by an expert to ensure correct values are chosen. Instead, this study employs a data-driven approach. In particular, to represent a dynamic system with a discrete state-space, the Hidden Markov Model (HMM) has shown to be an appropriate method and is also utilized by Wang *et al.* [45] in addressing an analogous issue. Especially, if the system's internal state is not accessible, but the system emits information that allows to infer about the hidden states. A HMM is fully characterized by the number of hidden states, the transition likelihood matrix and the parameter of the Gaussians or Gaussian Mixture Model (GMM) used for transmission modelling. The latter is dependent on the information emitted by the system. Since the distance to the lane marking is a continuous signal and used to infer the primitive, Gaussians are usually employed [84]. The transition matrix denotes the likelihood to transition from a state at the current time to another state in the next time step. This is useful in this use case, since the transitions

between the primitives are not equally likely. For instance, a transition from the state *Idle* to *Change* should be less likely than from *Idle* to *Approach*.

For using the HMM in this use case, the model parameters need to be defined. Instead of manually defining the transition probabilities between the states and the parameters of the Gaussians, we employ a data-drive approach and derive them from a collected dataset using the Baum-Welch algorithm [84] which also allows to apply the approach in different use cases. For demonstration, an exemplary drive on a motorway with a duration of fifteen minutes is selected which contains 39 lane-changes. The distribution of the normalized distance for that sequence is depicted in Figure 3.9. Noteworthy is, that the object in that sequence tend to drive on the left hand side of the lane since the distribution is not symmetrical around a distance of zero. In fact, the probability to drive on the right hand side is approximately 0.35 in this dataset. Although this might be the case only for this example, we will handle this asymmetric characteristic.

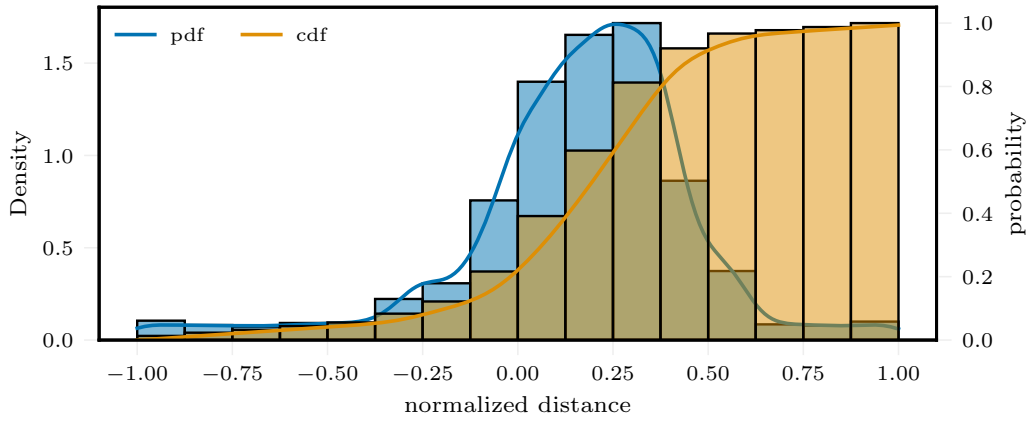


Figure 3.9: The distribution of the normalized distance for the example test drive. *Blue:* The probability density function. *Orange:* The cumulative distribution function.

As depicted in Figure 3.5 the domain consists of seven primitives. To overcome the potential problem of having an non-symmetrical distribution, we will drop the *left* and *right* variants for each primitive. That is, we will, at first, cluster the distribution into the four primitives

$$\mathbb{D} = \{\texttt{Idle}, \texttt{Approach}, \texttt{Cross}, \texttt{Change}\} \tag{3.12}$$

and recover direction for each primitive after the clustering. For that purpose, we use the absolute values of the context defined in (3.11) to estimate the HMM parameters using the Python library `hmmlearn`[4]. In Table 3.2 the estimated parameters for an HMM with Gaussian emissions is shown. That is, the transition probabilities between the states are given, whereas the primitives (hidden states) are represented as numbers. For every row, the probability to change to any other state is stated. The diagonal of the transition matrix denotes the probability to stay in the current state. Since the emissions are modelled with

[4] `https://github.com/hmmlearn/hmmlearn`

Table 3.2: The HMM parameter estimated from the example driving sequence with four internal states representing the primitives.

Primitive	Transition probability				Mean	
	0	1	2	3	d_c^c	κ
0	51.47	23.86	0.00	24.67	0.81 ± 0.04	1.00
1	17.23	82.14	0.00	0.63	0.95 ± 0.04	1.00
2	0.00	0.00	98.48	1.52	0.12 ± 0.08	0.00
3	1.00	0.08	1.82	97.11	0.39 ± 0.11	0.00

Gaussians, the components' means are given. Note that only the standard deviation for the normalized distance is depicted for visualization purpose, since the other covariances are close to zero which fits to our expectations for this use case.

Given the trained HMM we can transform the trajectory $\mathcal{X}$ into the domain of primitives using the Viterbi algorithm [84]. That is, if $\mathbf{k}_1, \ldots, \mathbf{k}_T$ is the trajectory in the context defined in (3.11), let $g\colon \mathbf{k} \to \mathcal{L}$ be the mapping of the context $\mathbf{k}$ to a primitive label from $\mathcal{L}$ with $\mathcal{L} = (0, 1, \ldots, |\mathbb{D}| - 1)$ as a tuple of primitive labels, the realization of g is the Viterbi algorithm. The result of applying the Viterbi algorithm to the example dataset is visualized in Figure 3.10. The distribution of the normalized distance d_c^c for each primitive is depicted as violinplots with each density function cut by the maximum value in the cluster. Note that the primitives are represented as numbers as in Table 3.2. The asymmetrical distribution is also visible in this figure especially for the first two primitives.

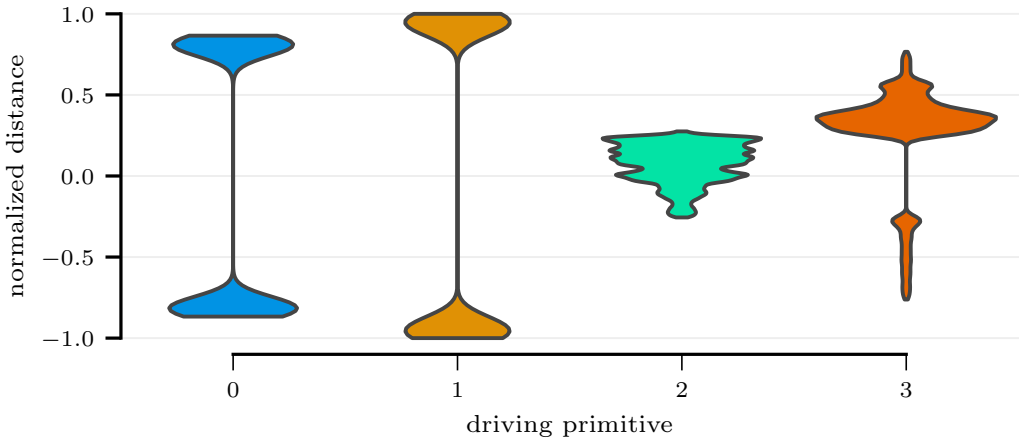


Figure 3.10: The distribution of the normalized distance for each primitive after applying HMM on the example test drive. The violinplot shows the density of a category. All clusters are bimodal except the primitive labeled with 2 which is unimodal.

The primitives in the HMM are represented as numbers. Since the emissions are modelled with Gaussians, which initial parameters are typically estimated using k-means, the

labels may change for different training runs. Due to that, the correct label of each primitive needs to be estimated, *i.e.*, we need to recover the correct semantic of each label. This is mandatory for follow-up steps in which we want to infer maneuvers based on their signatures defined in primitives domain. If the states of the HMM do not match to the primitives that we have defined, the maneuver signatures will not match as well. But, since those signatures are used for maneuver classification, we must ensure that the HMM states match with the primitive domain. Otherwise, robust maneuver classification is not possible.

To recover the correct semantic, we use the natural ordering of the primitive domain in terms of the lateral position of the vehicle in the lane. That is, let $\mu = \mu_0, \mu_1, \ldots, \mu_{|\mathbb{D}|-1}$ be the mean absolute normalized distances for the HMM's hidden states as depicted in Table 3.2, so that μ_i is the mean of the i^{th} hidden state and $i \in \mathcal{L}$ denotes the primitive label in Table 3.2. Let $\mathcal{L}^+$ be the correct labels, we seek finding a mapping $f : \mathcal{L} \rightarrow \mathcal{L}^+$ that relates every HMM label to the label giving the index of the correct primitive in $\mathbb{D}$ defined in (3.12). If we derive $\mathcal{L}^+$ by

$$\mathcal{L}^+ = \arg\operatorname{sort} \mu \tag{3.13}$$

with arg sort giving the index of the means μ sorted in ascending order, $f(i) = \mathcal{L}_i^+$ is a mapping between the primitive label i to the i^{th} component in $\mathcal{L}^+$. For this example, the tuple of correct primitive labels is $\mathcal{L}^+ = (2, 3, 0, 1)$ without specifying the direction of the primitive, so that, for instance $f(2) = 0$ would associate the HMM label 2 with the label 0. That is, the HMM state labeled with 2 would be related to the state `Idle` if it is used as an index for $\mathbb{D}$ from (3.12). This allows to review the HMM state transition matrix with the correct primitive. In Figure 3.11 the trained HMM with states represented in the primitive domain is illustrated. The nodes represent the HMM states and the connecting arrows the transitions between the states. For every transition, the probability is illustrated, while only transitions are shown with a probability greater than zero.

The fact that road users stay in a primitive for a longer period of time is also visible in the network as illustrated by the self referencing arrows in Figure 3.11. For instance, the probability $P(\texttt{Idle}|\texttt{Idle})$ of staying in the state `Idle` is 98.48% and the probability $P(\texttt{Approach}|\texttt{Approach})$ of staying in the `Approach` state is 97.10%.

If the road user starts in the state `Idle`, the typical path for a lane-change would be `Idle` $\rightarrow$ `Approach` $\rightarrow$ `Cross` $\rightarrow$ `Change`. However, the representation of the HMM as a network in Figure 3.11 uncovers an alternative path. If the road user is in the state `Approach`, there is a small chance to directly transition to the state `Change` and not visiting the state `Cross`. However, with a probability of $P(\texttt{Change} \mid \texttt{Approach}) = 0.08\%$ for the path from `Approach` to `Change` this is, compared to the usual path from `Approach` to `Cross` with a probability of $P(\texttt{Cross} \mid \texttt{Approach}) = 1\%$, quite unlikely. This alternative path also exists for the path from `Change` to `Approach`. But, this case is even more unlikely with a probability of $P(\texttt{Change} \mid \texttt{Approach}) = 0.63\%$ compared to the path $P(\texttt{Cross} \mid \texttt{Change}) = 17.22\%$.

The primitive domain in (3.12) lacks information about the lateral primitive direction, which is, however, mandatory to differentiate between a lane change to the left and right. To recover the directional information, the bimodal characteristic of each cluster's distribution (see Figure 3.10) is employed. That is, k-means is employed to partition the set

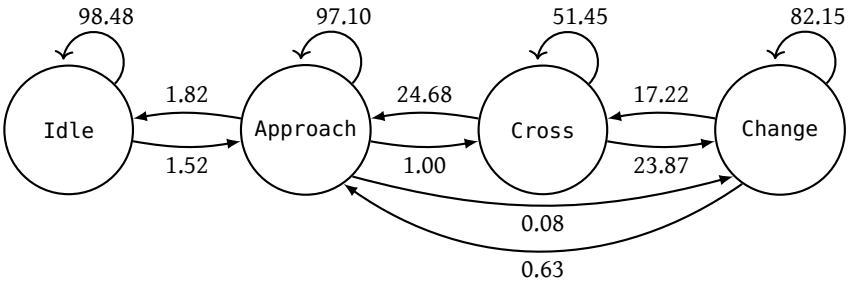


Figure 3.11: The HMM state transmission probabilities with state numbers replaced by the correct primitive.

of normalized distances for each primitive into two sub-clusters. That is, if μ_l^1, μ_l^2 are the means after clustering with k-means so that $\mu_l^1 > \mu_l^2$ and $l = f(g(\mathbf{k}))$ the label $l \in \mathcal{L}^+$ of the primitive, we can finally define the transformation function for representing the trajectory $\mathcal{X}$ into the domain of primitive $\mathcal{Y}$ using the primitive context with directional information as

$$y = t(\mathbf{k}) = \begin{cases} -l & \text{if} \quad l > 0 \wedge |\mathbf{k}^1 - \mu_l^1| \geq |\mathbf{k}^1 - \mu_l^2| \\ l & \text{else} \end{cases} \tag{3.14}$$

where $\mathbf{k}^1$ denotes the first component of the primitive context vector and thus the normalized distance d_c^c. Hence, the directional information of the primitives is encoded in the sign of the primitive label.

For the example drive, the normalized distance distribution after applying (3.14) is depicted in Figure 3.12. Each label on the ordinate is replaced with the correct primitive name, *i. e.*, the correct semantic is recovered. Furthermore, the samples of the primitives `Approach`, `Cross` and `Change` are further classified according to their direction as indicated by the different colors.

3.3.4 Maneuver classification based on primitives

So far, we have defined the context and domain for our use case of lane-change identification and derived a function that annotates a sample from a dataset of driving data with the primitive of the domain. Hence, the dataset can be represented in the domain of the context. Since the aim is to derive maneuvers from the primitives and identify those maneuvers in the dataset, we need to define the maneuvers $\mathcal{M}$ that should be identified. Furthermore, we have to identify the maneuvers in the data since multiple maneuvers may exist. These issues are addressed in the following.

For this case study, the aim is to identify lane-changes to the left and to the right lane. In the domain defined in (3.2) the maneuvers can be represented with the primitives as depicted in Figure 3.13. The figure shows the evolution of a vehicle performing a lane change from left to right. The top row shows a typical path for a left lane change with the

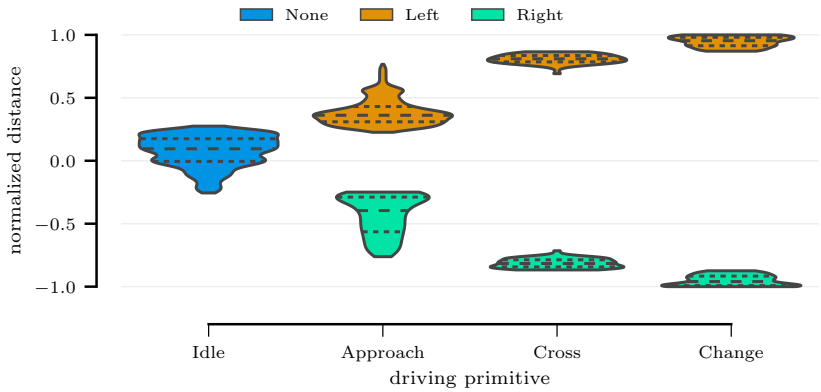


Figure 3.12: The distribution of the normalized distance for each primitive after clustering into the primitives of the domain defined for this context. The direction of a primitive is color-encoded and the labels are replaced with the name of the primitive.

destination lane being left of the initial lane. The bottom row of Figure 3.13 shows a typical path for a right lane change with the destination lane being right of the initial lane.

Remark that each primitive is represented with a label $y \in \{0, 1, 2, 3\}$ and the direction of a primitive is denoted in the sign of the label as defined in (3.14). Due to that, the signature of a lane-change to the left can be represented with the sequence of labels $m_1 = (0, 1, 2, 3, -3, -2, -1)$ and the signature of a lane-change to the right with $m_2 = (0, -1, -2, -3, 3, 2, 1)$ so that the set of maneuvers is defined as

$$\begin{aligned} \mathcal{M} &= \{\mathbf{m}_1, \mathbf{m}_2\} \\ &= \Big\{(0, 1, 2, 3, -3, -2, -1), \\ &\qquad (0, -1, -2, -3, 3, 2, 1)\Big\} \end{aligned} \tag{3.15}$$

for the identification a lane-change to the left and to the right.

The trajectory may contain multiple incidents of the maneuvers of interest. In fact, the example trajectory that is used in the previous section is depicted in Figure 3.14 as scatterplot in the domain of primitives, illustrating the potential existence of multiple maneuvers. Each sample of the dataset is represented as a transparent point to indicate the duration within a primitive. Overall, the vehicle is in the states `Idle` and `ApproachLeft` most of the time. But, it also leaves these states and even changes the lanes as denoted in the `ChangeLeft` and `ChangeRight` rows in Figure 3.14. Since the aim is to identify all maneuver incidents, the transition between the latter states are of special interest, while the information of the vehicle being in the state `Idle` is not relevant for lane-change identification. Hence, we will partition that dataset into sequences that contains those potential lane-change maneuvers and verify their existence afterwards using the maneuver signatures.

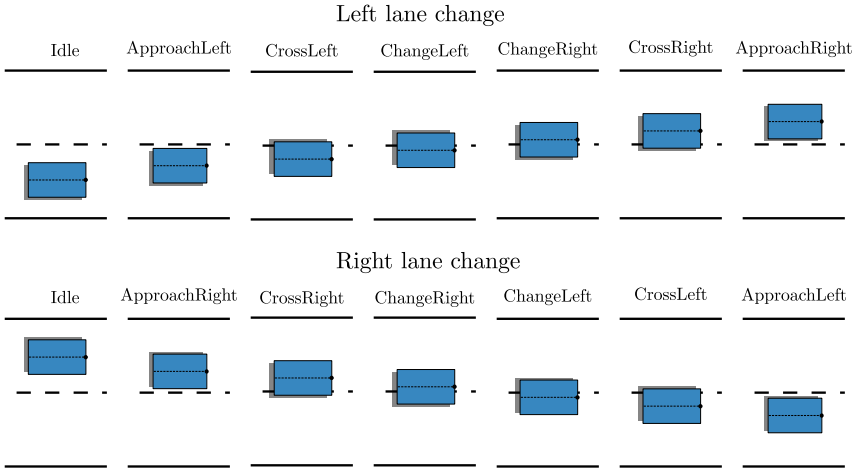


Figure 3.13: The lane-change to the left and right maneuvers represented as primitives from the defined domain.

For the partitioning of a trajectory based on the primitives, the absolute primitive label is employed. For the example lane-change maneuver, the signal of absolute primitive label is depicted in Figure 3.15. At the beginning of the sequence, the vehicle is in the state `Approach`. After approx. five seconds the vehicle moves towards the left lane and enters the state `Cross`, changes the lane until *2:11pm* where it reaches the `Approach` state again. The aim is to find that interval in the given sequence.

Since we are only interested in the sequences where the vehicle actually crosses the lane, we split up the dataset into sequences with all labels of that sequence greater than a defined minimum primitive label l_{min} and each sequence contains the primitives labels of the maneuver's signature. These properties of the sequences can be defined formally. That is, let $\mathcal{Y} = \mathcal{Y}_1, \mathcal{Y}_2, \ldots$ be the sequences of the trajectory in the primitive domain $\mathcal{Y}$ after partitioning. The following must hold for a sequence $\mathcal{Y}_k$

$$\forall y \in \mathcal{Y}_k, |y| \geq l_{min} \tag{3.16}$$

regarding the minimum primitive label l_{min} and

$$\forall l \in \mathbf{m}^* \, (\exists y \in \mathcal{Y}_k, y \equiv l) \tag{3.17}$$

regarding the existence of the primitive labels in the maneuver signature, with $\mathbf{m}$ as the maneuver's signature.

The result for the example lane-change by applying (3.16), (3.17) is depicted in Figure 3.16 for $l_{min} = 1$ and $l_{min} = 2$. Note that the selection of l_{min} impacts the length of the extracted sequence. For $l_{min} = 1$ the sequence also contains the interval before the actual lane-change, whereas a $l_{min} = 2$ only covers the interval where the vehicle transitions between the lanes.

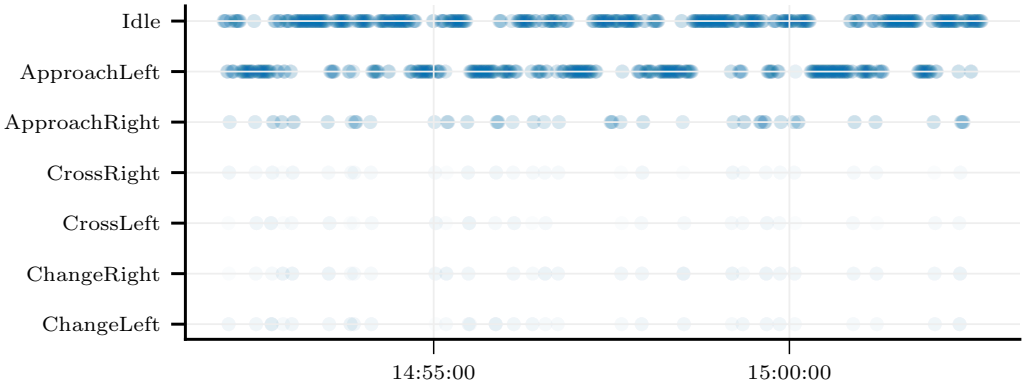


Figure 3.14: The example dataset represented in the domain of primitives for lane-change identification.

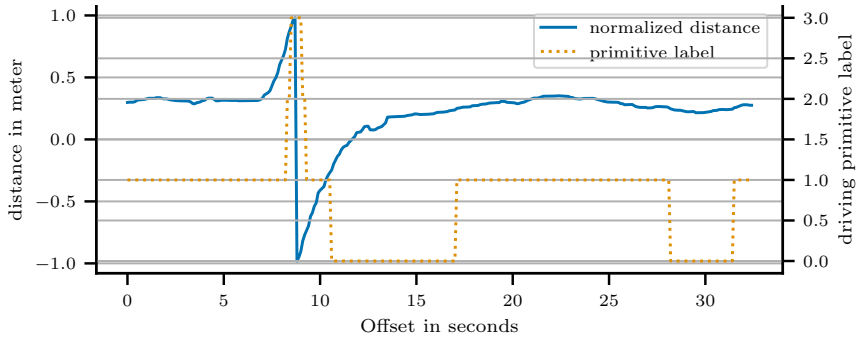


Figure 3.15: The example lane-change maneuver represented with the normalized distance and the absolute primitive label.

The trajectory $\mathcal{Y}$ is partitioned into the sequences $\mathcal{Y}_1, \mathcal{Y}_2, \ldots$, with each sequence $\mathcal{Y}_k$ potentially containing a maneuver signature. The final step is to infer the correct maneuver for each sequence, *i. e.*, perform a classification w.r.t the maneuvers of interest. Since every maneuver is defined in the domain of primitives by its signature, we can employ supervised-learning for maneuver classification. In the following, we use the minimum distance classifier to associate a sequence with one of the a priori defined maneuvers via their signatures. Formally defined, if $\mathcal{Y}_k$ is a sequence after partitioning and $\mathbf{m}^*$ the correct maneuver, the supervised classification problem using the minimum distance classifier is defined as

$$\mathbf{m}^* = \underset{\mathbf{m} \in \mathcal{M}}{\arg\min}\ \mathrm{d}(\mathcal{Y}_k, \mathbf{m}) \tag{3.18}$$

with $d(\mathcal{Y}_k, \mathbf{m})$ as a function giving the distance between the sequence $\mathcal{Y}_k$ in the primitive domain and the maneuver's signature $\mathbf{m}$. Choosing the correct distance function for this problem is, however, not trivial. The length of the maneuver signatures may not equal to the length of the sequence. Therefore, a variant of the Minkowski distance, *e. g.*,

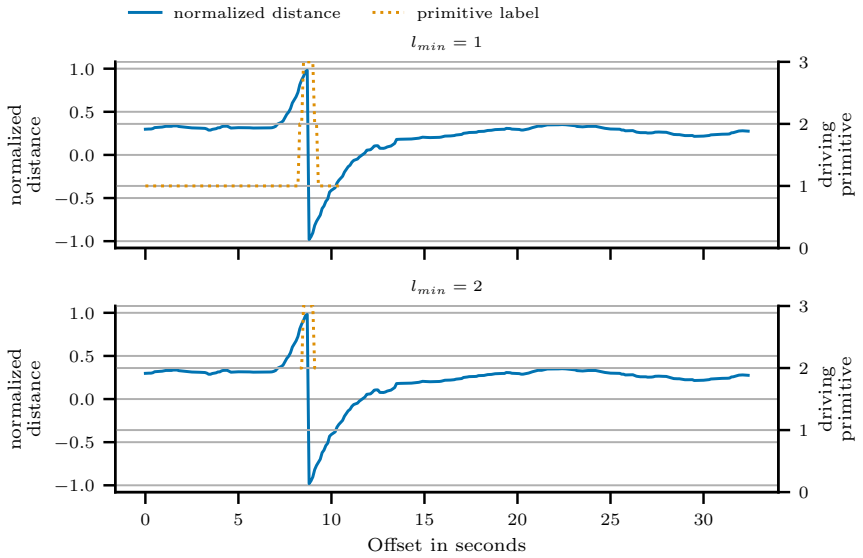


Figure 3.16: The partitioning result for the example lane-change maneuver. *Top:* The minimum absolute driving primitive label $l_{min} = 1$. *Bottom:* A minimum absolute primitive label $l_{min} = 2$. The selection of l_{min} defines the length of the extracted sequence.

the Euclidean distance or Manhattan distance, cannot be employed although it is usually used [81, p. 705]. Instead, if we view the maneuver signature as a time series we can employ methods for time-series comparison. Although different methods exist in literature for that task as discussed by [88], we propose to use DTW since it allows to compare two time series of different length by "both ignoring global and local shifts in the time dimension"[86] and it provides a *warped* variant of the time series. That is, it not only returns a distance between the two time series, but also a new time series aligned to the signature of the maneuver. An example lane-change sequence is illustrated in Figure 3.17 with the warped maneuver signatures $\mathbf{m}_1, \mathbf{m}_2$ and the sequence $\mathcal{Y}$ using DTW. The time series of $\mathbf{m}_1$ mostly matches with the sequence $\mathcal{Y}$, while the $\mathbf{m}_2$ only partially overlaps with it indicating that the sequence represents lane change to the left.

The partitioned sequences may include any of the maneuvers for which signatures where created. However, this must not always be the case. To ensure that the sequence really represents any of the maneuvers defined, an upper-limit distance d_{max} is specified using primitive labels. This distance is employed as threshold so that a sequence is only considered for further analysis if

$$\exists \mathbf{m} \in \mathcal{M},\ \mathrm{d}(\mathcal{Y}_k, \mathbf{m}) \leq d_{max} \tag{3.19}$$

holds for the sequence $\mathcal{Y}_k$ and the set of maneuvers $\mathcal{M}$. The maximum distance is derived from the sequence of primitive labels and the current primitive domain $\mathbb{D}$. That is, let $\mathcal{U}$ be a discrete uniform distribution of the domain of primitives and $X_t = |y_t - \mathcal{U}|$ a

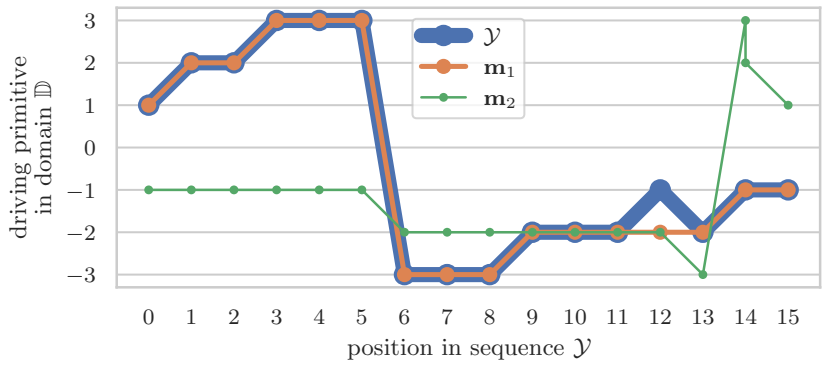


Figure 3.17: The warped maneuver signatures using DTW of $\mathbf{m}_1$ and $\mathbf{m}_2$ w.r.t the sequence $\mathcal{Y}$ in the domain $\mathbb{D}$ of primitives.

random variable as the absolute difference between the primitive $y_t \in \mathcal{Y}$ and the distribution $\mathcal{U}$. Then, the expected distance ϵ_t of X_t is defined as

$$\epsilon_t = E(X_t) = \sum_{l}^{\mathbb{D}} |(y_t - l)| \cdot P(\mathcal{U} = l) \tag{3.20}$$

where $P(\mathcal{U} = l)$ denotes the probability mass function of the discrete uniform distribution at each primitive l. The definition of the expected distance in (3.20) is employed to define the maximum distance d_{max} as

$$d_{max} = \sum_{t}^{T} \epsilon_t = \sum_{t}^{T} E(X_t) = \sum_{t}^{T} \sum_{l}^{\mathbb{D}} |(y_t - l)| \cdot P(\mathcal{U} = l) \tag{3.21}$$

with $y_t \in \mathcal{Y}$. That is, the maximum distance is the sum of the expected distances over the sequence $\mathcal{Y}$. The sum is used to ensure that d_{max} has the same scale as the result of DTW for proper comparison.

An example is depicted in Figure 3.18 for the lane-change to the left maneuver shown in Figure 3.16. The sequence was shorten for demonstration purpose to start and end in the primitive `Approach`.

The cell diagram at the bottom of Figure 3.18 depicts the manhattan distance between the primitive label of the sequence and every primitive of the domain per column. On the ordinate, the primitive labels of the sequence are shown. It should be noted that the vehicle can remain in a primitive for an arbitrary duration, justifying the use of DTW for time series comparison in this use case. The abscissa represents the primitive labels of the domain. The manhattan distance between the primitive in the sequence and the domain is color encoded from bright to dark. The top diagram of Figure 3.18 represents the distance between the warped maneuver signatures for the lane change to the left $\mathbf{m}_1$ and right $\mathbf{m}_2$ using DTW and the sequence. Moreover, the series of expected distances $\epsilon = \epsilon_1, \ldots, \epsilon_T$ is illustrated as a baseline. On the ordinate, the distance between the primitive label of the

sequence and the maneuvers are depicted. A small distance indicates a good match between the maneuver signature and the sequence, which is a necessary property for (3.18). For this example, the distances between the maneuver signatures and the sequence are $d(\mathbf{m}_1, \mathcal{Y}) = 1.0$, $d(\mathbf{m}_2, \mathcal{Y}) = 34.0$, whereas the maximum distance is $d_{max} = 39.14$. The only matching error of $glsManeuver_1$ occurs at the end of the maneuver, where the sequence changes from the primitive label -2 to -1 and back to -2. This can typically occur in real-world data, e.g., due to sensor measurement noise. Nevertheless, the property of (3.19) holds for this sequence and according to (3.18) the sequence $\mathcal{Y}$ is correctly classified as a left-turn maneuver.

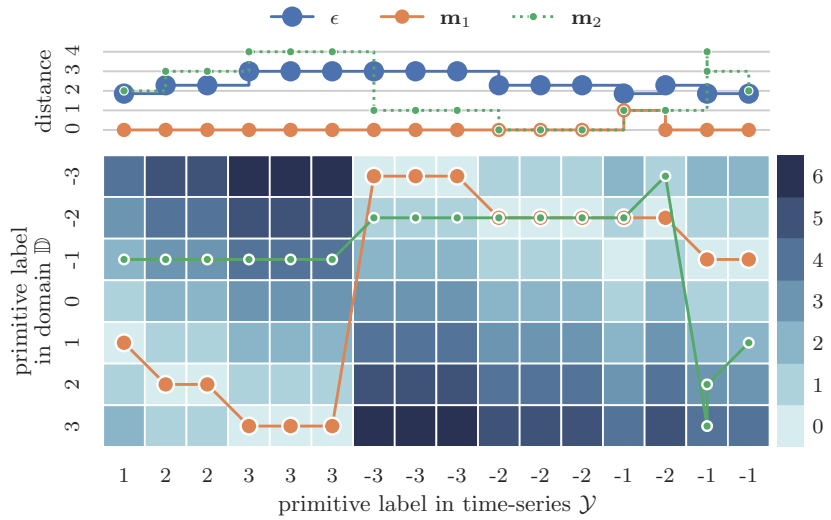


Figure 3.18: Visualization of matching distances using Dynamic Time Warping (DTW). The manhattan distance between each label in the domain $\mathbb{D}$ and the label in the sequence $\mathcal{Y}$ visualized as a cell plot (bottom) and the difference between the warped maneuvers as line plots (top) for $\mathbf{m}_1$ and $\mathbf{m}_2$ and the expected distance ϵ as a baseline reference.

For the example lane-change depicted in Figure 3.16 with different l_{min} the distance d_{max} is 218.14 for $l_{min} = 1$ and 56.57 for $l_{min} = 2$ due to the shorter sequence. The distances $d(s_1, Y)$ for both variants $l_{min} = (1, 2)$ are $(0.0, 2.0)$ and for $d(s_2, Y)$ are $(87.0, 51.0)$. Hence, that sequence is also correctly classified as left lane-change.

3.3.5 Results and discussion

In this chapter, a methodology for a driving-primitive based representation of driving data was proposed. The use case of lane-change identification based on primitives served as a proof-of-concept. To verify the robustness and accuracy of the approach for lane-change identification, an evaluation is conducted in the following. For that purpose, a dataset of manually annotated lane-changes is used and the extracted maneuvers by the proposed

method are compared to the ground truth lane-changes. The impact on the results of the domain verification step defined in (3.17) and the minimum primitive label l_{min} for partitioning are also analyzed.

Dataset

For the evaluation of the approach for lane-change identification, a dataset from the FASva project is used [89]. That dataset consists of a single trajectory on a motorway with a duration of 2.5 hours. In that dataset, the vehicle performs 205 lane-changes in total with 101 left and 104 right lane-changes. These lane-change were labelled manually by inspecting the images of the front-facing camera.

For each lane-change, the start and end times were annotated as shown in Figure 3.19, which is an example trajectory of a vehicle performing a left lane-change. In this work, the start of a lane-change is defined as that moment where the vehicle's heading changes towards the target lane. The end of the maneuver is in this work defined as that moment where the vehicle is on the target lane and the vehicle's heading is aligned to the lane's heading. In case of multiple consecutive lane-changes, the end of the first and start of the second maneuver is that moment where the vehicle is in the center of the lane.

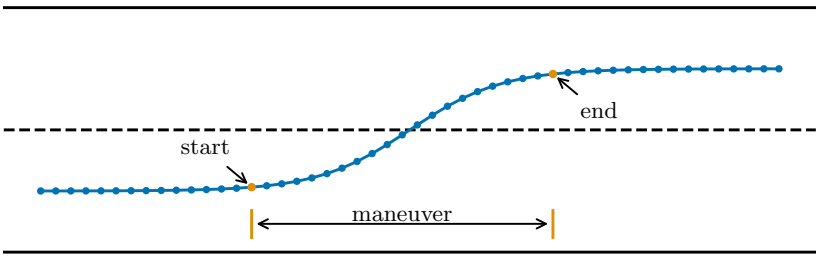


Figure 3.19: An example trajectory of an object performing a left lane-change. The manually annotated lane-change start and end times define the ground truth maneuver interval.

For the identification of lane-changes, the already derived parameters using the shorter sequence are used (see Table 3.2 for the HMM parameters). The interval of an extracted maneuver is determined by the sequence that is provided by the partitioning step described in Section 3.3.4 and is thus dependent on the l_{min} parameter. Due to that, a comparison of maneuver interval duration will also be conducted the following.

Metrics

For the quantitative analysis of the approach, the manually labelled set of lane-changes is compared to the set of lane-changes derived by the proposed method. By interpreting a lane-change maneuver as interval, the performance of the approach can be expressed quantitatively with metrics typically used to evaluate machine learning methods. That is,

a maneuver is correctly identified, a *true positive*, if the maneuver interval overlaps with the ground truth interval. A maneuver that is not associated to any manually labelled lane-change is marked as a *false positive*. All ground truth lane-changes having no associated lane-changes are *false negatives*. Those metrics can be combined into *precision* and *recall* [90] with the *precision* defined as

$$\text{precision} = \frac{\text{true positive}}{\text{true positive} + \text{false positive}} \tag{3.22}$$

and thus denoting the relation between the correctly identified lane-changes to the overall annotated lane-changes. That is, a *precision* of close to one denotes that all of the identified lane-changes really exist. However, that metric cannot be used alone since it does not contain information about how many of the actually available lane-changes in the dataset are found by the algorithm. For that purpose, the *recall* metric is used which is defined as

$$\text{recall} = \frac{\text{true positive}}{\text{true positive} + \text{false negative}} \tag{3.23}$$

and thus relating the correctly identified lane-changes to the total number of available lane-changes in the dataset. Both metrics defined in (3.22) and (3.23) can be combined into a single metric, the *f1-score* defined as

$$\text{f1-score} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.24}$$

which is used in the following for qualitative analysis.

The dataset partitioning into sequences according to the minimum primitive label l_{min} impacts the extracted maneuver intervals (see Figure 3.16). Due to that, the accuracy of the approach is also evaluated in terms of the alignment of the correctly extracted to the manually annotated maneuvers. An example is depicted in Figure 3.20 showing a left lane-change with the manually annotated interval *reference* and the *extracted* interval. In this work, the accuracy of maneuver interval extraction is related to the overlap between the extracted interval to the reference interval. As for the lane-change identification analysis, the *precision*, *recall* and *f1-score* metric can also be employed for evaluation. Therefore, let the *true positive* (tp) be the number of samples in the extracted interval that are also part of the reference interval. The *false positives* (fp) are the samples in the extracted interval that are not part of the reference interval. All samples of the reference interval that are not part of the extracted interval are considered as *false negatives* (fn).

Results

The results of lane-change identification are shown in Table 3.3. To study the effect of the domain verification step defined in (3.17) and the dataset partitioning on the performance of lane-change identification, the results for different combination are shown. For the dataset partitioning, l_{min} is chosen as either one or two since with a minimum label of $l_{min} = 0$ the dataset will not be partitioned and the sequences using $l_{min} = 3$ would

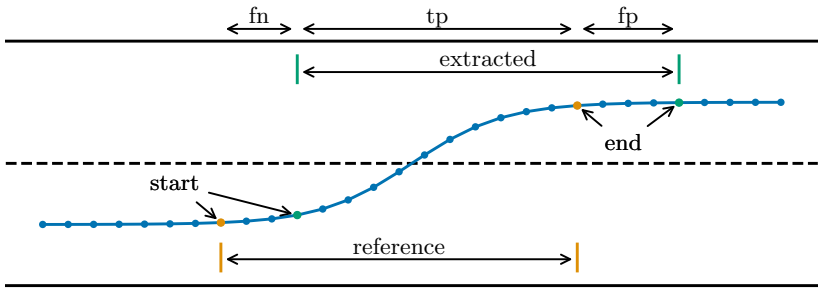


Figure 3.20: An example trajectory of an object performing a left lane-change with the manually annotated and the extracted maneuver interval.

represent only the transitioning phase of the lane-change. The rows denoted with *verified* represent the results with and *unverified* without domain verification.

The experiment shows that partitioning the sequence according to a minimum primitive label significantly affects the overall performance. A sequence partitioned with a primitive label of 1 (the state `Approach`) results in fewer correctly identified lane change maneuvers than with a primitive label of 2 (the state `Cross`). Moreover, the likelihood increases of extracting lane change maneuvers that do not occur. This is indicated with a slight reduction of *precision* for $l_{min} = 1$. The domain verification mitigates this problem and ensures that only valid maneuvers are extracted. This more robust extraction comes at the cost of fewer maneuvers being extracted, as indicated by a slight decrease of *recall*. Those invalid sequence lack some of the primitives. That this can potentially occur was already indicated by the trained HMM transmission probabilities depicted in Figure 3.11. Due to the bidirectional connection between the primitives `Approach` and `Change`, there is a chance to skip the state `Cross` at all. Thus, if the sequence does not contain the state `Cross`, it will not be considered for classification.

In summary, the combination of $l_{min} = 2$ and no domain verification provide the highest *f1-score* of 0.988. That is, only five lane-changes were not identified leading to a *recall* of 0.976, whereas all extracted lane-changes are correct as denoted with zero false positives and a *precision* of 1.000.

	l_{min}	TP	FN	FP	Precision	Recall	F1-score
unverified	1	173	32	11	0.940	0.844	0.889
	2	200	5	0	1.000	0.976	**0.988**
verified	1	167	38	0	1.000	0.815	0.898
	2	197	8	0	1.000	0.961	0.980

Table 3.3: The evaluation results of lane-change identification with and without domain verification and different minimum primitive labels for partitioning.

The sequence partitioning using a minimum primitive label affects the length of the sequences. To investigate whether the extracted maneuvers using the two minimum labels $(1, 2)$ match with a human-extracted (reference) lane change interval, the duration and the extracted intervals are analyzed with the manually created reference intervals. In Figure 3.21 the duration distribution of the correctly extracted and their associated reference maneuvers are depicted for $l_{min} = 1$ and $l_{min} = 2$ with domain verification. The duration probability density function of $l_{min} = 2$ is significantly denser around the mean duration of 2.08 ± 0.89 seconds as compared to $l_{min} = 1$ with a duration of 6.71 ± 7.18. In summary, lane change maneuvers extracted with a primitive label of 1 tend to be longer than with a primitive label of 2. This is compatible with our expectations since maneuvers extracted using a lower minimum primitive label represent more states of the lane change maneuver.

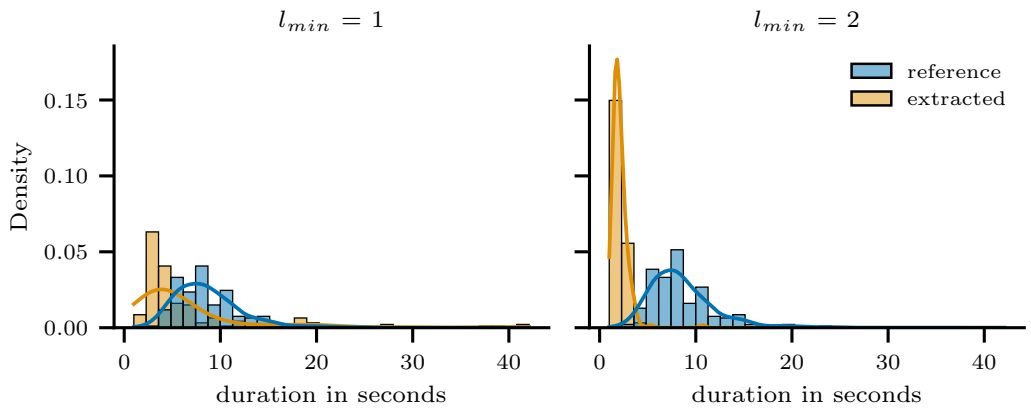


Figure 3.21: The distribution of the extracted and reference lane-change maneuvers. *Left:* The result for partitioning with $l_{min} = 1$. *Right:* The distribution after partitioning with $l_{min} = 2$.

The experiments also indicate that the extracted maneuvers tend to be shorter than the reference maneuvers. The Table 3.4 shows the mean duration and the standard deviation for the correctly extracted and their associated reference lane changes. There is a significant difference of approx. 4.4 seconds between the mean duration for $l_{min} = 2$ and their reference lane-changes having a mean duration of 8.469 ± 3.287 seconds. This deviation is closer for $l_{min} = 1$ with a mean duration difference of approx. 1.9 seconds compared to the reference lane change. But, there is a higher variation of maneuver lengths with a standard deviation of 7.186 which is more that two times higher than the deviation of the reference maneuver durations. This could occur because of a vehicle staying in the state `Approach` for a longer period. While the extracted maneuver includes this phase, the reference maneuver may only contain the actual maneuver. Thus, maneuvers extracted with different primitive label represent different aspects of the actual lane change.

To investigate what phases of the actual lane change the extracted intervals represent, the maneuvers in light of the Figure 3.20 are analyzed. The distribution on a per lane

	l_{min}	duration in seconds mean	std
extracted	1	6.718	7.186
	2	2.085	0.890
reference		8.662	3.428

Table 3.4: The mean duration and standard deviation of correctly extracted lane-changes and their associated reference lane-changes.

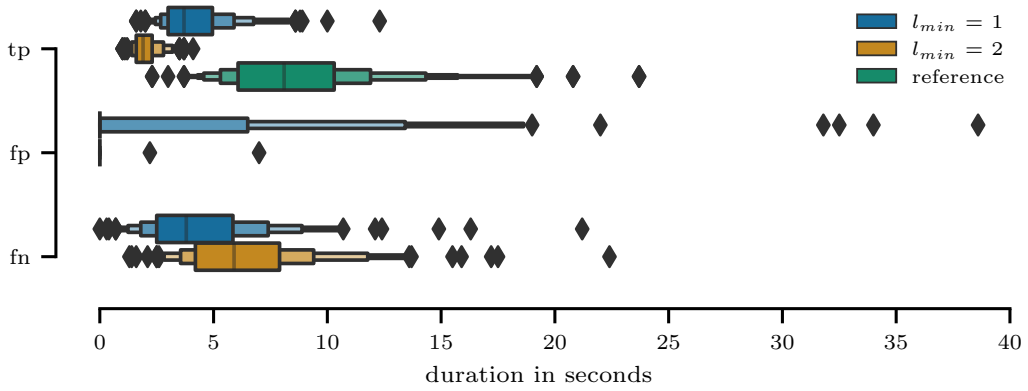


Figure 3.22: The distribution of the correctly extracted lane-change parts for $l_{min} = 1$, $l_{min} = 2$ and the *reference* lane-changes for comparison. Each lane change is split up into *tp*, *fp* and *fn* according to Figure 3.20.

change basis is illustrated in Figure 3.22 as letter-value plots, with the duration distribution of the reference lane changes depicted in green in the *tp* row for comparison. The experiment shows that maneuvers extracted with $l_{min} = 1$ have a significantly higher overlap with the reference lane changes as with $l_{min} = 2$ (see *tp* row). This comes at the cost of including parts of the trajectory that are not in the reference lane-changes (see *fp* row). That is, although the medians of $l_{min} = 1$ and $l_{min} = 2$ for *fp* are both close to zero, using $l_{min} = 1$ leads to a higher number of outliers. Moreover, due to the fact that intervals extracted with $l_{min} = 1$ tend to be longer than with $l_{min} = 2$, they tend to contain a higher portion of the actual maneuver (see *fn* row). This is also depicted in Figure 3.23 showing the *Recall*, *Precision* and *F1-score* based on (3.23), (3.22) and (3.24). In this context, *recall* denotes the fraction of the correctly extracted lane-change interval compared to the reference lane-change. That is, the higher the *Recall*, the more of the actual lane-change is part of the extracted one. *Precision*, in turn, relates the fraction of the extracted lane-change that is part of the actual lane-change to the overall extracted lane-change. A higher *Precision* thus means that the extracted interval has a high overlap with the actual lane-change. The higher amount of outliers for $l_{min} = 2$ is present in the *Precision* row. Furthermore, Figure 3.23 highlights the overall higher overlapping between the extracted and reference

lane-changes with a significantly higher *Recall* for $l_{min} = 1$. This also has an impact on the overall performance of lane-change extraction as represented by the *f1-score* with a mean *f1-score* $= 0.591 \pm 0.149$ for $l_{min} = 1$ and *f1-score* $= 0.405 \pm 0.109$ for $l_{min} = 2$.

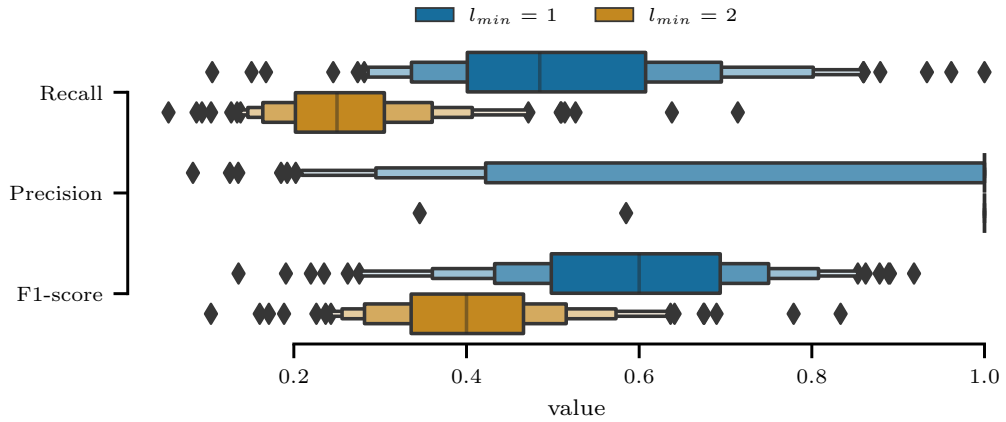


Figure 3.23: The evaluation results of the extracted lane-change intervals compared to the reference lane-changes denoting the overlap between extracted and reference lane-changes

An example outlier visible in the *Precision* row in Figure 3.23 as well as in the *fp* row in Figure 3.22 is depicted in Figure 3.24. That extracted lane change contains approx. 35 seconds that are not part of the reference lane-change (see rightmost marker in *fp* of Figure 3.22). The Figure 3.24 shows the normalized distance d, absolute normalized distance $|d|$ and the primitive label. The reason for this high deviation between the reference lane-change and the extracted lane-change is because of the vehicle staying in the `Approach` state after the lane-change. Since the vehicle's heading recovers to the target lane after the actual lane-change, the end of the maneuver was, however, already annotated after the peak close to the moment where the vehicle enters the `Approach` state. Although this is arguable since the vehicle changes to the `Idle` state after staying for this period of approx. 35 seconds in the `Approach` state, it is consistent with the definition of the end of a maneuver used in this work.

In summary, it was shown that robust maneuver identification is possible by representing a lane change maneuvers with primitives and classify maneuvers by defining their signatures in the domain of primitives. Moreover, the experiment shows that maneuver extraction based on the primitives has shown to be sensitive to the sequence partitioning. A more complete representation of the reference maneuver (with $l_{min} = 1$) comes at the cost of including parts of the trajectory that are not part of the actual maneuver. In contrast, extracting maneuvers with the aim of having sequences representing a part of the actual maneuver (with $l_{min} = 2$) limits the ability to capture the entire maneuver. These properties must be taken into account for the respective application.

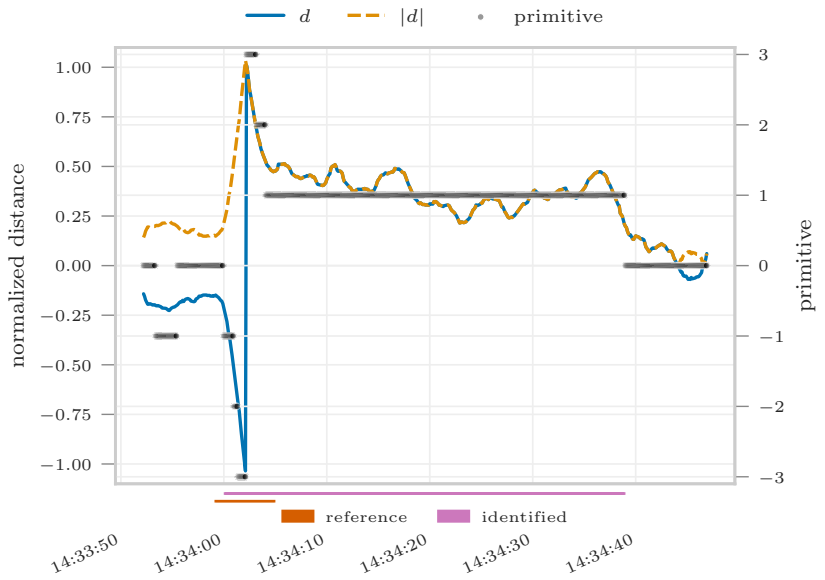


Figure 3.24: An example lane-change extracted with $l_{min} = 1$ showing the sensitivity of the label-based dataset partitioning method. The right part of the identified lane-change overshooting the actual lane-change has a duration of approx. 35 seconds.

3.4 Case study: Route estimation on an urban intersection

The following study will demonstrate the approach of representing real-world driving data with primitives for the use case of classifying traffic participants driving route on an urban intersection represented using the OpenDRIVE format. That is, it will be shown how to estimate the route that motorized road users follow on an urban intersection using the presented approach and combined with the results of the approach we will present in Section 4, how to classify traffic participant's routing maneuver on urban intersections. The overall approach is illustrated in Algorithm 1, which we will present in the following[5].

[5]Parts of the presented approach are published in [89].

Algorithm 1 Route estimation on an urban intersection.

Input: Trajectory $\mathcal{X}$, domain $\mathbb{D}$
Output: Route P

1: $D \leftarrow$ Initialize an instance of DAGMaR
2: **for all** pose $\mathbf{x}_t$ of trajectory $\mathcal{X}$ **do**
3: $r_1, r_2, \ldots, r_{|\mathcal{V}|} \leftarrow$ roads in the vicinity $\mathcal{V}(\mathbf{x}_t) \subset \mathbb{D}$ of the pose $\mathbf{x}_t$
// Perform map matching of the current pose
4: $\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_{|\mathcal{V}|}^* \leftarrow$ projected poses of $\mathbf{x}_t$ to roads $r_1, r_2, \ldots, r_{|\mathcal{V}|}$
// Compare the object pose with the projected poses
5: $\mathcal{L}(\mathbf{x}_1^* \mid \mathbf{x}_t), \mathcal{L}(\mathbf{x}_2^* \mid \mathbf{x}_t), \ldots, \mathcal{L}(\mathbf{x}_{|\mathcal{V}|}^* \mid \mathbf{x}_t) \leftarrow$ likelihood of the projected poses
// Iteratively build DAGMaR
6: update DAGMaR D with roads $r_1, r_2, \ldots, r_{|\mathcal{V}|}$, the projected poses $\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_{|\mathcal{V}|}^*$ and their mapping likelihood $\mathcal{L}(\mathbf{x}_1^* \mid \mathbf{x}_t), \mathcal{L}(\mathbf{x}_2^* \mid \mathbf{x}_t), \ldots, \mathcal{L}(\mathbf{x}_{|\mathcal{V}|}^* \mid \mathbf{x}_t)$
7: **end for**
8: $P \leftarrow$ The longest path in DAGMaR D from the most likely road at $r_{t=1}$ to $r_{t=|\mathcal{X}|}$

3.4.1 Primitive domain

To represent the traffic participant in the domain of primitives, it first needs to be defined. In this study, it is assumed that, if a road user is on a road connecting two intersections, it is straight forward to identify the road that the traffic participant is on. Hence, the focus is on route estimation when the traffic participants crosses an intersection. Thus, the domain of primitives is defined w.r.t the intersections of that road network and all other roads are neglected. For instance, the domain of primitives for the intersection depicted in Figure 3.25 is the set of incoming, connecting and outgoing roads. In OpenDRIVE, every road has a unique number for identification purpose. Hence, the domain of primitive is defined as

$$\mathbb{D} = \left\{ r_1, r_2, \ldots, r_N \right\} \tag{3.25}$$

and represents the set of all roads that traffic participants use to cross the intersection with $r \in \mathbb{D}$ as a unique road number.

3.4.2 Primitive context

To associate a road user's pose to a primitive, or road in this use case, we employ *map matching*. There exists various variants in literature for this process. A typical approach is to use the location and evaluate with which road of a network the road user is associated to. We will follow this strategy and use the road user's location and orientation information. Since we will evaluate this association for all poses, we perform a road search (see Algorithm 1:3) in the proximity of the road user's current pose and project the pose onto all roads (see Algorithm 1:4). Those projected poses will be used to compare the road user's pose against the projected ones.

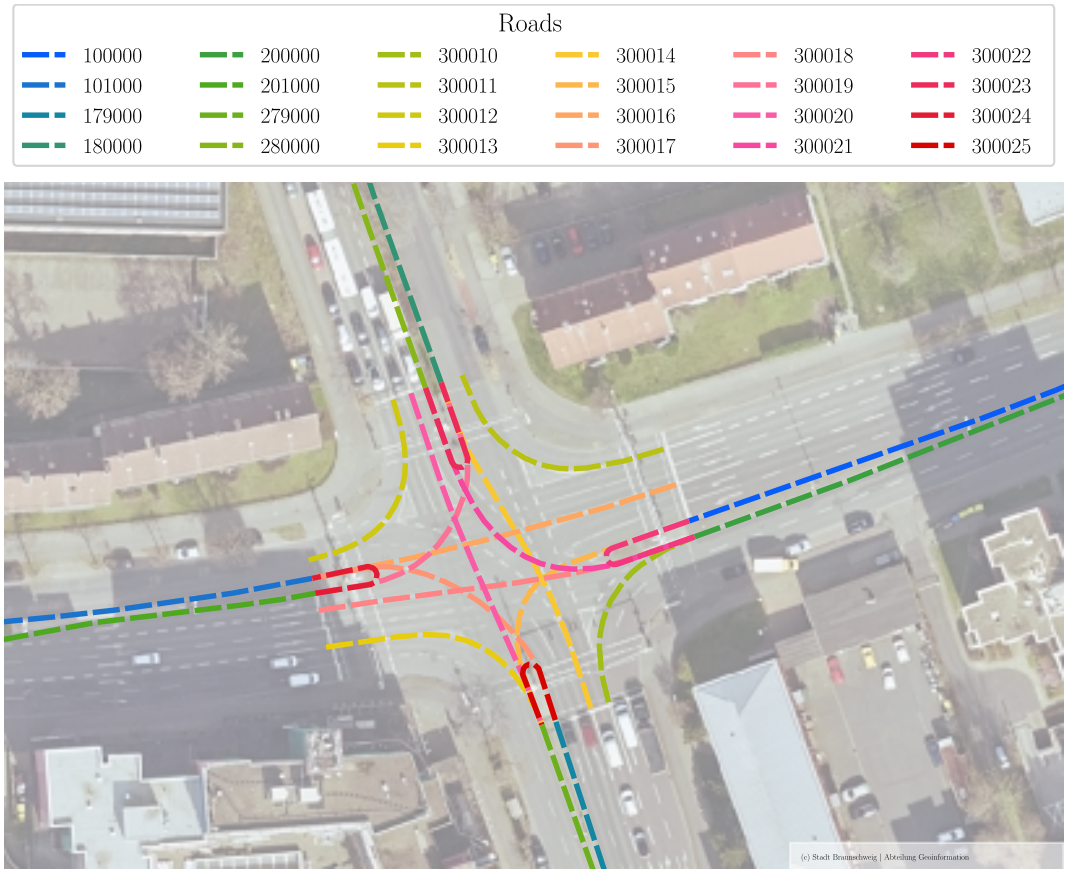


Figure 3.25: The DLR AIM research intersection and the reference lines of its digital representation in OpenDRIVE format.

The context for this use case is defined by the orientation deviation and distance between the road user's pose and the projected poses as illustrated in Figure 3.26. The left panel shows the trajectory of the road user and two roads represented by their reference line. The right panel shows the road user's pose $\mathbf{x}_t$, the projected poses $\mathbf{x}_1^*$, $\mathbf{x}_2^*$ and the context. Formally defined, let a pose $\mathbf{x}_t$ of the road user's trajectory $\mathcal{X}$ be redefined based on (2.2) as

$$\mathbf{x}_t = \begin{bmatrix} \mathbf{p} & \omega \end{bmatrix}^T \tag{3.26}$$

at time t, with the position denoted as $\mathbf{p} = [u, v]^T$ in UTM coordinate system (see Section 2.1.2). The orientation is given as $\omega \in (-\pi, \pi]$ with $\omega = 0$ pointing east, increasing in counter-clockwise direction. Furthermore, let $\mathcal{X}^*$ be the trajectory of projected poses $\mathbf{x}_t$ analogous to (2.3) defined as

$$\mathcal{X}^* = \left(\mathbf{x}_1^*, \mathbf{x}_2^*, \ldots, \mathbf{x}_{|\mathcal{V}|}^*\right) \tag{3.27}$$

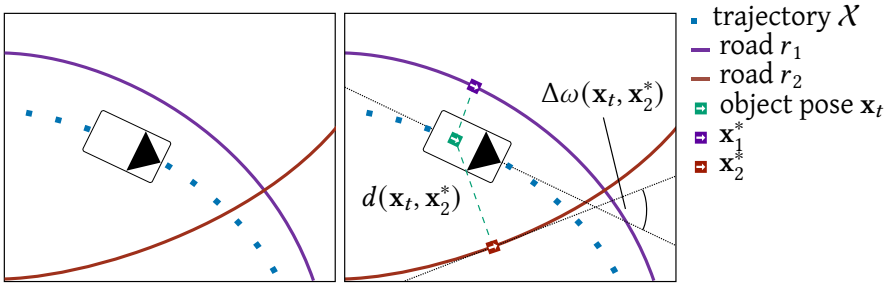


Figure 3.26: The context for route estimation using map matching. *Left:* The object's trajectory with roads r_1, r_2 in the vicinity represented by their reference lines. *Right:* The object pose projected on both roads $(\mathbf{x}_1^*, \mathbf{x}_2^*)$ with the context illustrated for the pose $\mathbf{x}_t$ projected on the road r_2 via pose $\mathbf{x}_2^*$.

such that $\mathbf{x}_k^* = [\mathbf{p}, \omega]^T$ is the road user's pose projected onto the road r_k in its vicinity $\mathcal{V}$. The vicinity $\mathcal{V}(\mathbf{x}_t) = \{r_1, r_2, \dots\}$ is a set of roads within the proximity of the pose $\mathbf{x}_t$. Then, the position deviation $d(\mathbf{x}_t, \mathbf{x}_k^*)$ between the road user's pose $\mathbf{x}_t$ and the projected pose $\mathbf{x}_k^*$ is defined by the Euclidean distance

$$d(\mathbf{x}_t, \mathbf{x}_k^*) = \left\| \begin{bmatrix} u_{\mathbf{x}_t} \\ v_{\mathbf{x}_t} \end{bmatrix} - \begin{bmatrix} u_{\mathbf{x}_k^*} \\ v_{\mathbf{x}_k^*} \end{bmatrix} \right\| = \left\| \mathbf{p}_{\mathbf{x}_t} - \mathbf{p}_{\mathbf{x}_k^*} \right\| = \sqrt{\sum^{i} \left(\mathbf{p}_{\mathbf{x}_t}^i - \mathbf{p}_{\mathbf{x}_k^*}^i \right)^2} \tag{3.28}$$

or the Euclidean norm of the difference between the positions $\mathbf{p}_{\mathbf{x}_t}$ and $\mathbf{p}_{\mathbf{x}_k^*}$ for the road user's and the projected pose. This is illustrated in the right panel of Figure 3.26 as the green dashed line between the object pose $\mathbf{x}_t$ and the projected pose $\mathbf{x}_2^*$ on road r_2.

To estimate the orientation deviation $\Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*) \in (-\pi, \pi]$ we transform the orientation of $\mathbf{x}_t$ and $\mathbf{x}_k^*$ from polar to cartesian coordinates such that $(\mathbf{u}, \mathbf{v})$ are the unit vectors in Cartesian coordinates. Then, the orientation deviation is defined as

$$\Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*) = \Delta\omega(\mathbf{u}, \mathbf{v}) = \operatorname{arctan2}\Big(\det([\mathbf{uv}]), \mathbf{u}^T\mathbf{v} \Big) \tag{3.29}$$

with $\det([\mathbf{uv}])$ the sine, $\mathbf{u}^T\mathbf{v}$ the cosine of the angle between them and arcan2 the angle between the point and the ordinate to ensure a four-quadrant and signed solution since the road's unit vectors can lie arbitrarily in $\mathbb{R}^2$. This angle is illustrated in the right panel of Figure 3.26 exemplarily for the object pose $\mathbf{x}_t$ and the projected pose $\mathbf{x}_2^*$ as $\Delta\omega(\mathbf{x}_t, \mathbf{x}_2^*)$. The context for this use case is thus defined as

$$\mathcal{K} = \Big\{ \mathbf{k} \Big\} = \Big\{ (d, \Delta\omega) \Big\} \tag{3.30}$$

by the Euclidean Distance d and the orientation deviation $\Delta\omega$.

3.4.3 Transformation of road user trajectory

To estimate the route of a traffic participant, the trajectory is transformed into the domain of primitives using the context that was defined in the previous section. As stated earlier, the aim is to find arbitrary maneuvers and do not limit the number of maneuvers beforehand by defining their signatures, which was the case for the previous study. Instead, we will represent the way how the road user's cross the intersection via roads in a subset of the overall intersection's roads. For that purpose, a Directed Acyclic Graph (DAG) is utilized that allows to represent roads as nodes and the transition between roads as arcs. The roads in the DAG are named in the following as *mapped roads* and the special variant of the DAG as Directed Acyclic Graph of Mapped Roads (DAGMaR). Moreover, since an object can be mapped to multiple roads in the current vicinity, every projected pose is weighted to ultimately find the route.

For that purpose, let $\mathcal{L}(r_1 \mid \mathbf{x}_t), \mathcal{L}(r_2 \mid \mathbf{x}_t), \ldots, \mathcal{L}(r_{|\mathcal{V}|} \mid \mathbf{x}_t)$ be the mapping likelihoods of the roads in the vicinity $\mathcal{V}(\mathbf{x}_t)$, *i. e.*, the likelihoods of the road user's pose $\mathbf{x}_t$ being projected on the roads in the vicinity. Since the context is defined by the Euclidean distance $d(\mathbf{x}_t, \mathbf{x}_k^*)$ and the orientation deviation $\Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)$ between the pose $\mathbf{x}_t$ and the projected pose $\mathbf{x}_k^*$ of the k^{th} road $r_k \in \mathcal{V}$, the likelihood of the pose $\mathbf{x}_t$ being projected onto r_k or the mapping likelihood of r_k is defined as

$$\mathcal{L}(r_k \mid \mathbf{x}_t) = \frac{\mathcal{L}\big(r_k \mid d(\mathbf{x}_t, \mathbf{x}_k^*)\big)\mathcal{L}\big(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)\big)}{\sum_i^{\mathcal{V}} \mathcal{L}\big(r_i \mid d(\mathbf{x}_t, \mathbf{x}_i^*)\big)\mathcal{L}\big(r_i \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_i^*)\big)} \tag{3.31}$$

and thus the joint likelihood of $\mathcal{L}\left(r_k \mid d(\mathbf{x}_t, \mathbf{x}_k^*)\right)$ and $\mathcal{L}\left(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)\right)$. The denominator in (3.31) ensures that $\sum_r^{\mathcal{V}} \mathcal{L}(r_k \mid \mathbf{x}) = 1$. For both likelihood functions $\mathcal{L}\left(r_k \mid d(\mathbf{x}_t, \mathbf{x}_k^*)\right)$, $\mathcal{L}\left(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)\right)$ we use models that satisfy requirements for this use case, which we will define in the following.

For the distance likelihood model $\mathcal{L}(r_k \mid d(\mathbf{x}_t, \mathbf{x}_k^*))$, the following requirements are defined. The OpenDRIVE reference lines are used as a geometric representation of roads (as illustrated in Figure 3.26). Since they represent the leftmost border of the road, its width needs to be integrated into the model. Furthermore, the model should allow to integrate uncertainty of road width estimation and support the fact that traffic participants might not follow the digital representation of the road exactly but leave it in, for instance, curves. Due to that, the Tukey window [91] was adapted as

$$\mathcal{L}\left(r_k \mid d(\mathbf{x}_t, \mathbf{x}_k^*)\right) = \begin{cases} \epsilon + \frac{1}{2}(1-\epsilon)\left[1 + \cos\left(\pi\frac{d}{b}\right)\right] & \text{for} \quad -b \leq d < 0 \\ 1 & \text{for} \quad 0 \leq d < a \\ \epsilon + \frac{1}{2}(1-\epsilon)\left[1 + \cos\left(\pi\frac{d-a}{c}\right)\right] & \text{for} \quad a \leq d < (a+c) \end{cases} \tag{3.32}$$

which is a piecewise function with three parameters. The parameter a denotes the deviation from the reference road for the right side and thus can be used to define the road's width. The parameters (b, c) allow to model the deviation of the traffic participant from

the road for the left side and right side. The parameter $\epsilon \ll 1$ defines the minimum likelihood.

In this work, two different parameter configurations are employed, one for turning roads and one for straight roads (see Figure 3.27). For turning roads, traffic participants are allowed to deviate from the road, while the likelihood decrease is dependent on the turning direction. For instance, the top plot of Figure 3.27 illustrates the likelihood distribution for a single lane left-turning road with $a = 3.75$ meter. In this case, the deviation magnitude to the left side is equal to the road's width and to the right side of the road is two times the road's with. For straight roads, the allowed deviation is smaller as for turning roads (see bottom plot of Figure 3.27). In this case, the deviation to the left and to the right is depended on the lane width and not on the overall road width, which is the case for turning roads. In particular, we generously assume a road width of $w = 3.75$ meter for german roads[6] and that $b = c = \frac{1}{2}w$ for straight roads.

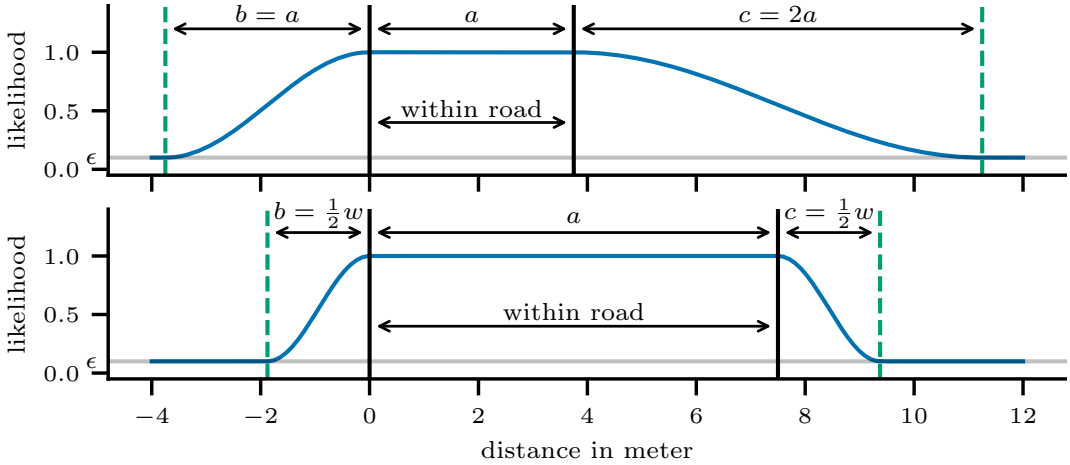


Figure 3.27: The adapted Tukey window used as the likelihood function to model the road user's distance to the road with lane width of $w = 3.75$ meter. *Top:* The model for a single lane left-turning road. *Bottom:* The model for a straight road with two lanes.

For the orientation deviation model, the following requirements are defined. At first, the likelihood needs to be inversely proportional to the orientation deviation. That is, the likelihood of being projected on a road is high for low orientation differences and decreases with increasing deviation. Furthermore, the relationship between the orientation deviation increase and likelihood decrease should be non-linear to stronger penalize higher orientation deviations. Finally, the minimum likelihood given as a model parameter should be at $|\pi|$ since $\Delta\omega \in (-\pi, \pi]$, so that

$$\min \mathcal{L}\left(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)\right) = \mathcal{L}\left(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*) = |\pi|\right) = \epsilon \tag{3.33}$$

[6] The maximum road widths for inner-city roads is defined in [92] as 3.25 meter and for motorway roads in [93] as 3.75, while is used in this work.

with $\epsilon \in (0, 1], \epsilon \ll 1$ defining the minimum likelihood at $|\Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)| = \pi$. This work employs an exponentially decaying function to model the orientation deviation likelihood, satisfying the defined criteria with

$$\mathcal{L}(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)) = \max\left\{\epsilon, \epsilon^{\frac{|\Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)|}{\pi}}\right\} \tag{3.34}$$

as depicted in Figure 3.28 with different values of ϵ in linear (top) and logarithmic scale (bottom). It is clearly visible that changing ϵ impacts the penalization of orientation deviations with a high penalization for smaller values of ϵ. The equations (3.32), (3.34) plugged into (3.32) allow to estimate the likelihood of the mapped roads in Algorithm 1:5.

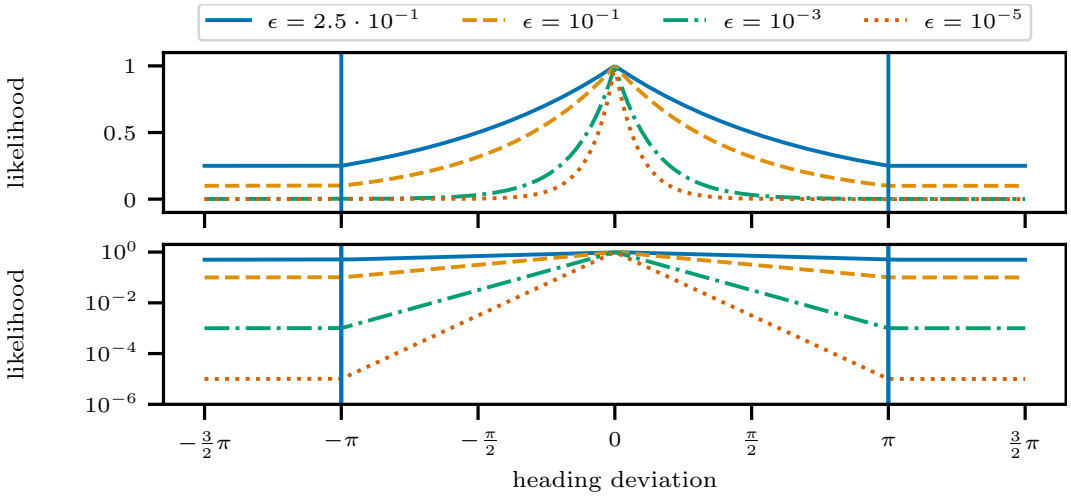


Figure 3.28: The exponential decaying function used as the likelihood function to model the road user's orientation deviation from a road. *Top:* The likelihood for different parameters. *Bottom:* The likelihood in log-space depicting that $\min \mathcal{L}(r_k \mid \Delta\omega(\mathbf{x}_t, \mathbf{x}_k^*)) = \epsilon$.

The projected poses, the road mapping likelihoods and the roads of the current vicinity are employed to update DAGMaR in line 6. In a nutshell, if a pose is projected onto a road that is not mapped in DAGMaR and that road is connected with any other road that is mapped in DAGMaR, it will be added as successor of that road. If a pose is projected onto a road that is already mapped in DAGMaR, the weight of that road node in DAGMaR will be incremented by the likelihood of the projected pose. It is worth noting that the road's geometric representation is used to check whether two roads are connected. That is, if two roads intersect, they are considered connected. This allows to find arbitrary routes that road users follow to cross the intersection, not only routes that are allowed and thus represented in the OpenDRIVE map. After processing the trajectory $\mathcal{X}$, the weight w_i of a primitive or road r_i in DAGMaR D is then given by

$$w_i = \sum_{\mathbf{x}}^{\mathcal{X}} \mathcal{L}(r_i \mid \mathbf{x}), \quad \forall r_i \in D \tag{3.35}$$

as the sum of the road mapping likelihood for all poses.

An example showing the evolution of DAGMaR is depicted in Figure 3.29 for five time steps and three roads. In the first time step t_0, the road user is mapped to only a single road r_0, for instance, before entering the intersection. In the three following time steps t_1, t_2, t_3 the vehicle is mapped to r_0 and r_1 with varying likelihood, *e.g.*, while crossing the intersection. In this situations, the vehicle could be close to multiple roads or could be transitioning from the current road to the next one. In the last step t_4 the road user is mapped onto r_1 and r_2, while the road r_2 is the successor of both r_0 and r_1. In this step, the road user might leave the intersection via road r_2.

The example illustrates that the transformation function used for this use case is a transformation of the trajectory to a DAG of primitives. That is, we employ an intermediate representation of the trajectory as a DAG. This allows us to represent a pose of the trajectory with *multiple* weighted primitives, the relationship between the primitives and the history of the visited roads.

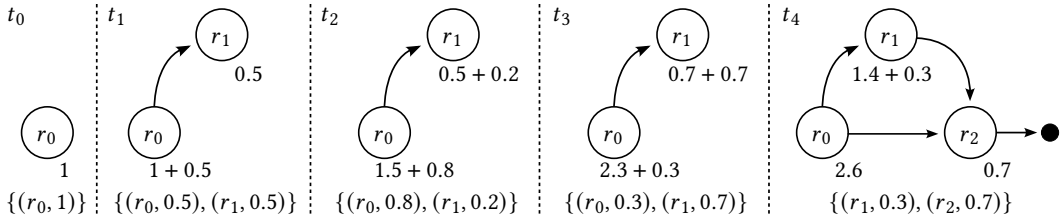


Figure 3.29: An example to demonstrate the evolution of DAGMaR for five time steps and three roads. The final route of the road user is the path with highest overall weight.

3.4.4 Routing maneuver estimation

The final step to estimate the routing maneuver given the representation of the road user's trajectory as a DAGMaR is to find the road user's route in DAGMaR. For that purpose, it is assumed that the road user will enter the intersection using an incoming road and leave the intersection via an outgoing road. Since those roads connect intersections in a OpenDRIVE network, it is assumed that road association is straight forward. Hence, the road user can be unambiguously associated to these roads at the start or end of the trajectory.

Formally defined, let (r_0^*, r_N^*) be the associated roads for the first pose of the trajectory r_0^* and the last pose of it r_N^*. Moreover, let us assume that there exists multiple paths $\mathcal{P} = \{P_1, P_2, \dots\}$ from r_0^* to r_N^* in DAGMaR. We seek finding the longest path P^* defined as

$$P^* = \arg\max_{P} \sum_{i=1}^{|P|} w_i \tag{3.36}$$

with $|P|$ as the number of roads of the path P and w_i the mapping likelihood of the road $r^* \in P$ as defined according to (3.31). For the representation of DAGMaR and for working with DAGs in Python we use the NetworkX Python package [94]. It allows to find all paths $\mathcal{P}$ from r_0^* to r_N^* using a recursive depth-first search from [95]. Hence, we can easily solve (3.36) to find the road user's path P^*.[7] For the example depicted in Figure 3.29, the routes from r_0 to r_2 would be $\{P_1 = (r_0, r_1, r_2), P_2 = (r_0, r_2)\}$. The sum of weights for both paths are $(5, 3.3)$. Therefore, P_1 is the longest path in DAGMaR D and the road user's route for crossing the intersection.

3.4.5 Results and discussion

The presented approach of representing a road user's trajectory in the domain of primitives for the purpose of route estimation on intersections will be validated in the following. Since the map matching based approach uses likelihood models to estimate the road mapping likelihood, the approach will be exemplarily demonstrated using trajectories of road users performing a *left turn*, *right turn*, *u-turn* and *straight ahead* maneuver on the DLR AIM Research Intersection. The four traffic participant's trajectories are depicted in Figure 3.30 (solid lines) with the OpenDRIVE road network (dashed lines), which roads define the primitive domain for this use case. Unless otherwise noted, the trajectories have been down sampled to a frequency of one Hertz, which is still a high-frequency sampling rate compared to other, albeit GPS-based, datasets [96].

We will compare the results of the presented approach with two baseline approaches. The first `MLCR` is based on the assumption that in an OpenDRIVE map, a direction for crossing an intersection is represented by a specific road, which we will be referred to in the following as connecting roads. We will select the path that connects an incoming road to an outgoing road via the most likely connecting road r^* defined as

$$r^* = \arg\max_r \sum_{\mathbf{x}}^{\mathcal{X}} \mathcal{L}(r \mid \mathbf{x}), \quad \forall r \in W_{\text{connect}} \tag{3.37}$$

with W_{connect} as the set of connecting roads. The second method `MLMP` is a common approach for route finding, where the path $P = \{r_1^*, r_2^*, \dots\}$ is estimated based on the most likely mapped road per time r_t^* defined as

$$r_t^* = \arg\max_r \mathcal{L}(r \mid \mathbf{x}_t), \quad \forall r \in \mathcal{V}_t \tag{3.38}$$

with $\mathcal{V}_t$ as a set of roads in the vicinity of the road user's pose $\mathbf{x}_t$ at time t and $\mathcal{L}(r \mid \mathbf{x}_t)$ the mapping likelihood as defined in (3.31).

[7] It is worth noting that an alternative solution to find the longest paths between two nodes in an acyclic graph, is, according to [95], to negate the weights and search for the shortest path using Floyd's algorithm. For the sake of clarity and since we assume that the number of possible routes is small, we use the exhaustive search approach.

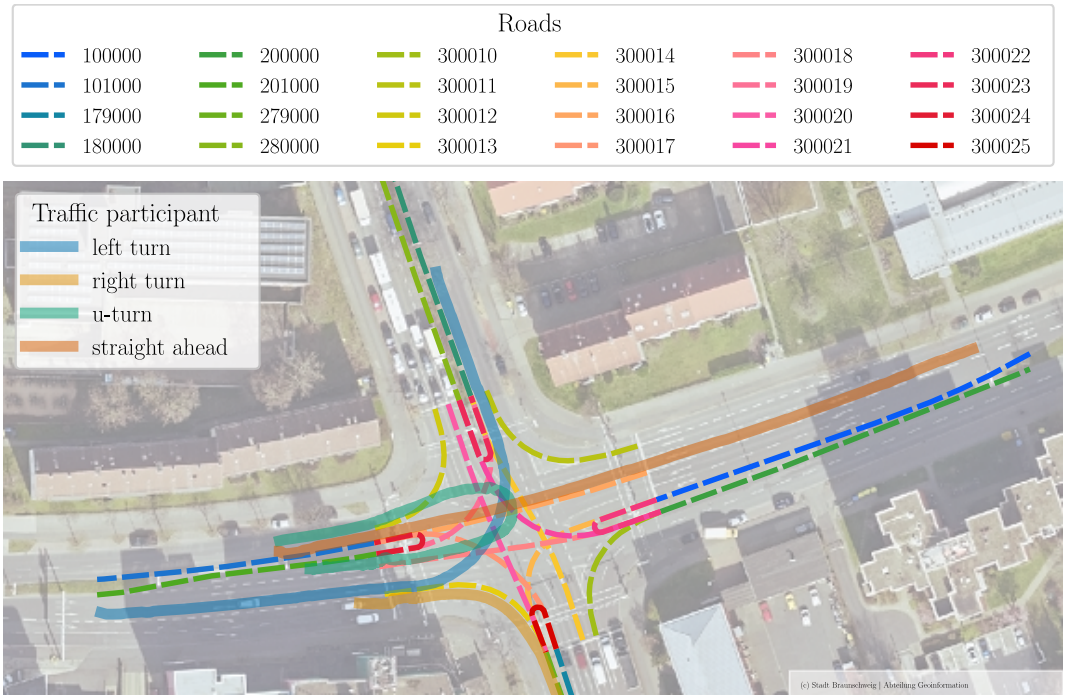


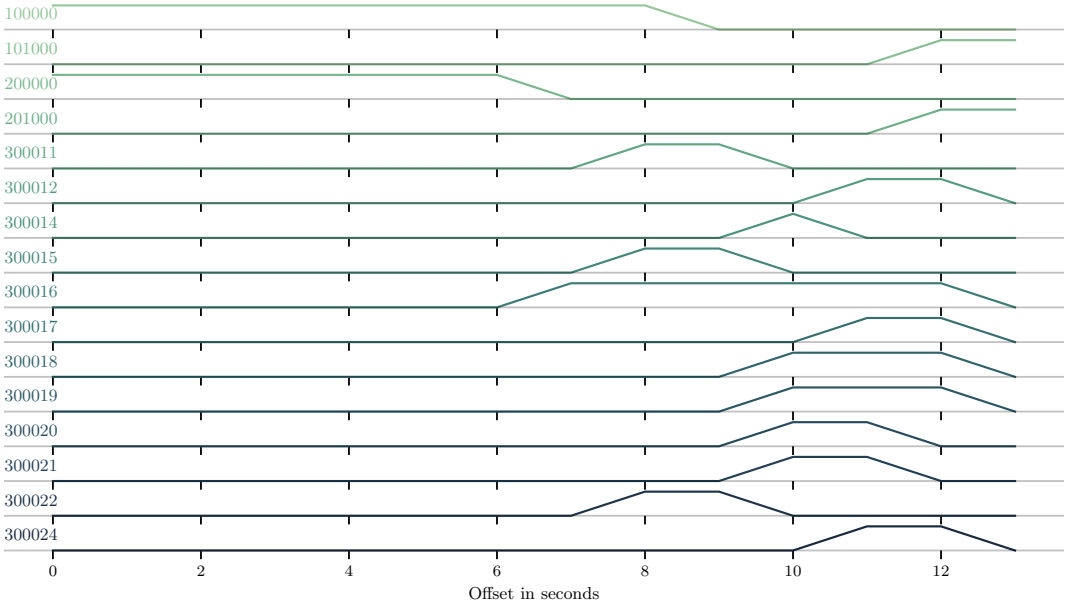
Figure 3.30: The trajectories of the traffic participants and the roads of the digital network representing the DLR AIM research intersection to demonstrate the primitive based route estimation approach.

Straight ahead

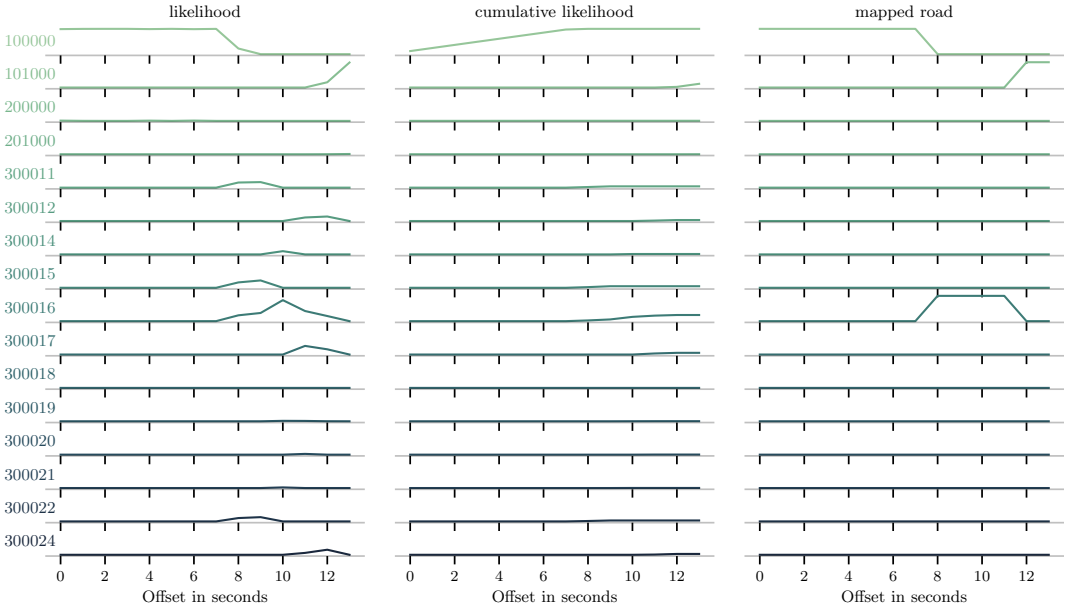
At first, we will discuss the results for the traffic participant performing a *straight ahead* maneuver. The road user enters the intersection from east and leaves it into west. While driving through the intersection, the road user's crosses various roads of the network, *i. e.*, the roads within the traffic participant's vicinity changes. Figure 3.31a illustrates the evolution of the vicinity over the course of the trajectory. Every road is represented by a binary signal. If the road is in the vicinity at the given time instant, the signal is high, otherwise it is low. Until t_7[8], the traffic participant approaches the intersection, which is indicated by a vicinity defined by the incoming road 100000 and the outgoing road 200000. The road user crosses the intersection from t_7 until t_{12}, because at t_{13} the vicinity contains only the outgoing road 101000 and incoming road 201000. The total duration for crossing the intersection is 5 seconds. Throughout the trajectory, 16 of all 24 roads (66.66%) were within the vicinity at least one time. The correct path of the road user is {100000, 300016, 101000}.

For both baseline approaches and the proposed one, the road mapping likelihood is used to find the taken route of a traffic participant. Figure 3.31b illustrates the mapping likelihood and the cumulative mapping likelihood per road. Moreover, the most likely mapped way per time and way is each depicted as a binary signal, which indicates that the traf-

[8]Note that we will refer to specific time instants using t_n, where n is the offset in seconds.



(a) The roads in the vicinity of the traffic participant.



(b) The (cumulative) mapping likelihood and the most likely mapped road per time.

Figure 3.31: The road search and mapping results for the traffic participant performing a *straight ahead* maneuver.

fic participant is mapped onto that road if the signal is high. Hence, it is used by `MLMP` for route estimation. In fact, the path by `MLMP` is $P_{\texttt{MLMP}} = \{100000, 300016, 101000\}$ for the *straight ahead* maneuver.

The MLCR approach constructs that path by finding the most likely connecting road. This is illustrated in Figure 3.31b as the cumulative likelihood. That is, the connecting road having the highest cumulative likelihood at the latest time is chosen for route estimation. For the *straight ahead* maneuver, the most likely connecting road is 300016 and thus the path $P_{\text{MLCR}} = \{100000, 300016, 101000\}$ using the topology of the intersection.

The result of the proposed approach uses a DAG of primitives (DAGMaR) for route estimation. For the *straight ahead* maneuver, the evolution of DAGMaR is illustrated in Figure 3.32 for four different times (t_8, t_8, t_{10}, t_{12}). The left panel of Figure 3.32 illustrates the trajectory and the four poses of the traffic participant with the road search radius. The right panel shows the state of DAGMaR as a graph at the specific times. The result of MLCR for each time while updating DAGMaR is shown as the highlighted route and the underlined road is the most likely connected road. The final path found via DAGMaR is $P_{\text{DAGMaR}} = \{100000, 300016, 101000\}$ as illustrated with the highlighted path in Figure 3.32 at t_{12}. In this case, all three variants estimate the correct path.

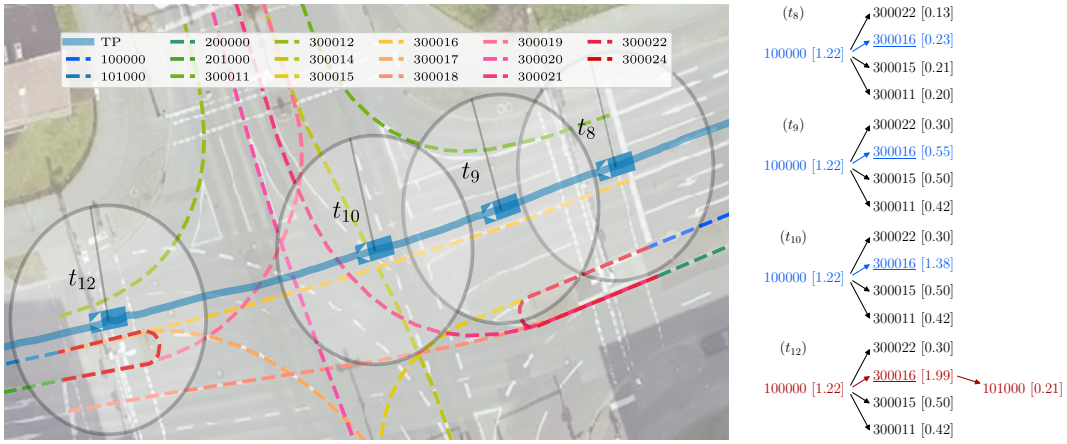


Figure 3.32: Demonstration of the driving-primitive based route estimation approach using a traffic participant performing a *straight turn* maneuver on the intersection. *Left:* The trajectory of the traffic participant and the poses for four time instants t_8, t_9, t_{10}, t_{12} with the road search radius illustrated that is used to find the road in the vicinity. *Right:* The evolution of the primitive graph (DAGMaR) for the four time instants, with roads of the intersection represented by their unique identifier and their cumulative mapping likelihood in square brackets. The most likely connected road is underlined, while both, the most likely path during updating DAGMaR and the final path by DAGMaR are highlighted.

Left turn

The traffic participant that performs a *left turn* maneuver enters the intersection from west and leaves it into north. It is worth noting that the traffic participant does not use the left turn lane. Instead, it uses the rightmost lane of the two straight ahead lanes (see Figure 3.30). Although this scenario is unusual, we use it as an example to challenge the

different approaches. As in the previous section, the roads within the traffic participant's vicinity changes. Figure 3.33a illustrates the vicinity over the course of the trajectory. Until t_5, the traffic participant approaches the intersection, which is indicated by a vicinity defined by the incoming road 101000 and the outgoing road 201000. The road user enters the intersection at t_5 and leaves it at t_{13}, because at t_{14} the vicinity contains only the outgoing road 18000 and incoming road 280000. Hence, the total duration for crossing the intersection is 8 seconds. Throughout the trajectory, 15 of all 24 roads (62.5%) were within the vicinity at least one time. The correct path of the road user is $\{201000, 300019, 180000\}$.

For both baseline approaches and the proposed one, the road mapping likelihood is used to find the taken route of a traffic participant. Figure 3.33b illustrates the mapping likelihood, the cumulative mapping likelihood per road and the most likely mapped road per time and road as a binary signal. The estimated path by MLMP for the *left turn* maneuver is $P_{\text{MLMP}} = \{201000, 300018, 300019, 300023, 180000\}$.

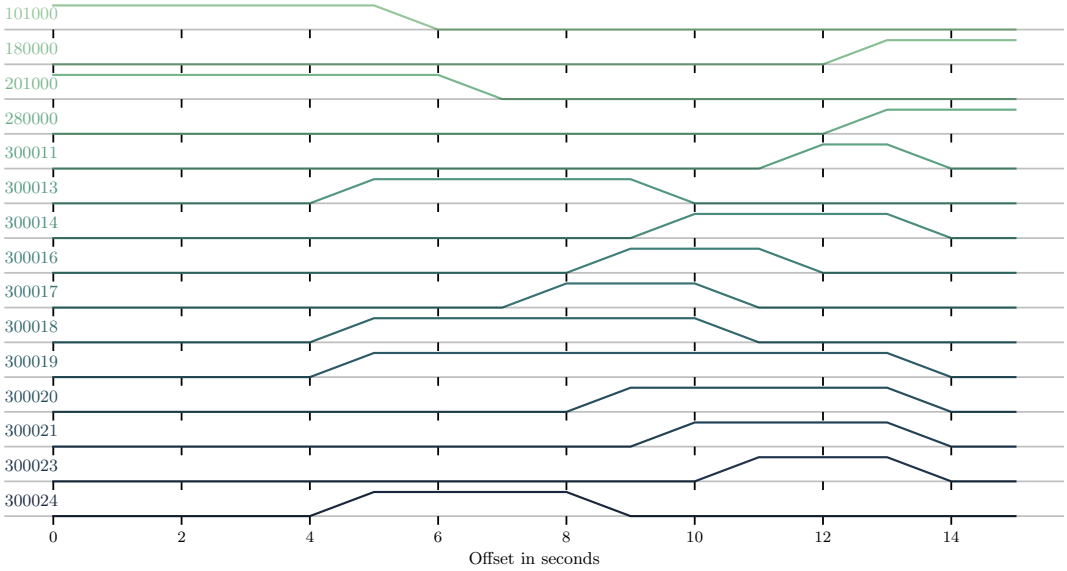
The MLCR approach constructs that path by finding the most likely connecting road. This is illustrated in Figure 3.33b as the cumulative likelihood. For the *left turn* maneuver, the most likely connecting road is 300019. By using the topology of the intersection, the path is $P_{\text{MLCR}} = \{100000, 300019, 180000\}$.

The result of the proposed approach uses a DAG of primitives (DAGMaR) for route estimation, which evolution is illustrated in Figure 3.34 for four different times (t_5, t_8, t_{10}, t_{13}) analogous to the *straight drive* maneuver. The final path found via DAGMaR is $P_{\text{DAGMaR}} = \{201000, 300019, 180000\}$ as illustrated with the highlighted path in Figure 3.34 at t_{12}. It is worth noting that the road 300023 which is part of the path P_{MLMP} is not part of DAGMaR since it is not connected to any road within DAGMaR. Hence, the topology of the intersection ensures that the traffic participant is only mapped onto a *valid* path. Since the road user's path to cross the intersection is $\{201000, 300019, 180000\}$, MLCR and the proposed method estimate the correct path.

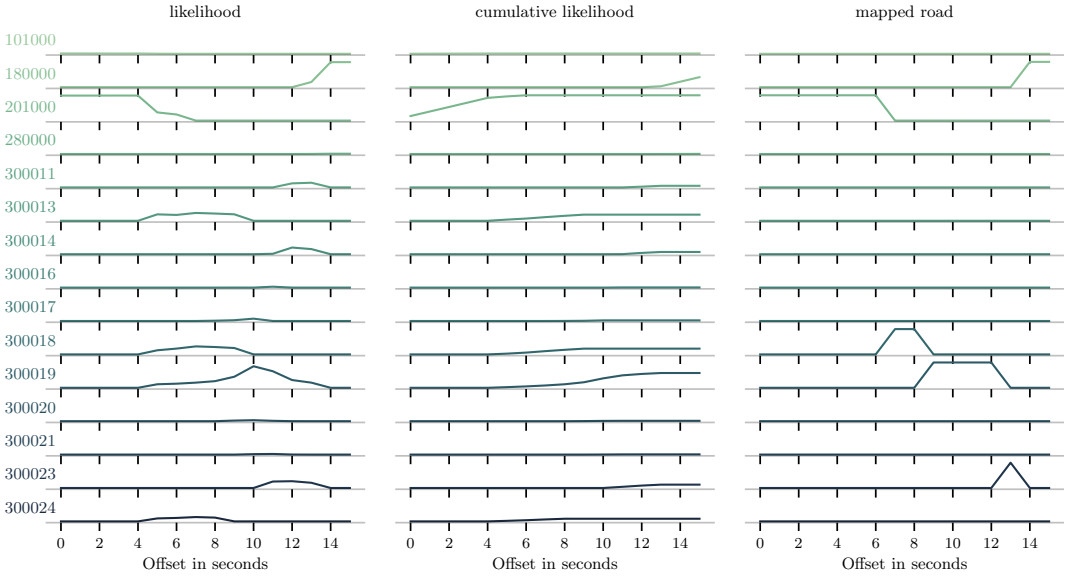
Right turn

The next results we will discuss is for the traffic participant that performs a *right turn* maneuver. The road user enters the intersection from west and leaves it into south. As in the previous section, the roads within the traffic participant's vicinity changes, which is illustrated in Figure 3.35a. In this case, the road user enters the intersection at t_1, since the vicinity at t_1 is defined by the incoming road 201000 and the connecting roads 300013, 300018. Note that there is no outgoing road at t_1 as opposed to the previous two examples. The road user leaves it at t_9, because at t_{10} the vicinity contains only the incoming road 179000 and outgoing road 279000. Hence, the total duration for crossing the intersection is 9 seconds. Throughout the trajectory, 9 of all 24 roads (37.5%) were within the vicinity at least one time. The correct path of the road user is $\{201000, 300013, 279000\}$.

For both baseline approaches and the proposed one, the road mapping likelihood is used to find the taken route of a traffic participant. Figure 3.35b illustrates the mapping likelihood, the cumulative mapping likelihood per road and the most likely mapped road per time and road as a binary signal. The estimated path by MLMP for the *right turn* maneuver is $P_{\text{MLMP}} = \{201000, 300013, 279000\}$.



(a) The roads in the vicinity of the traffic participant.



(b) The (cumulative) mapping likelihood and the most likely mapped road per time.

Figure 3.33: The road search and mapping results for the traffic participant performing a *left turn* maneuver.

The MLCR approach constructs that path by finding the most likely connecting road. This is illustrated in Figure 3.35b as the cumulative likelihood. For the *right turn* maneuver, the most likely connecting road is 300013. By using the topology of the intersection, the path is $P_{\texttt{MLCR}} = \{201000, 300013, 279000\}$.

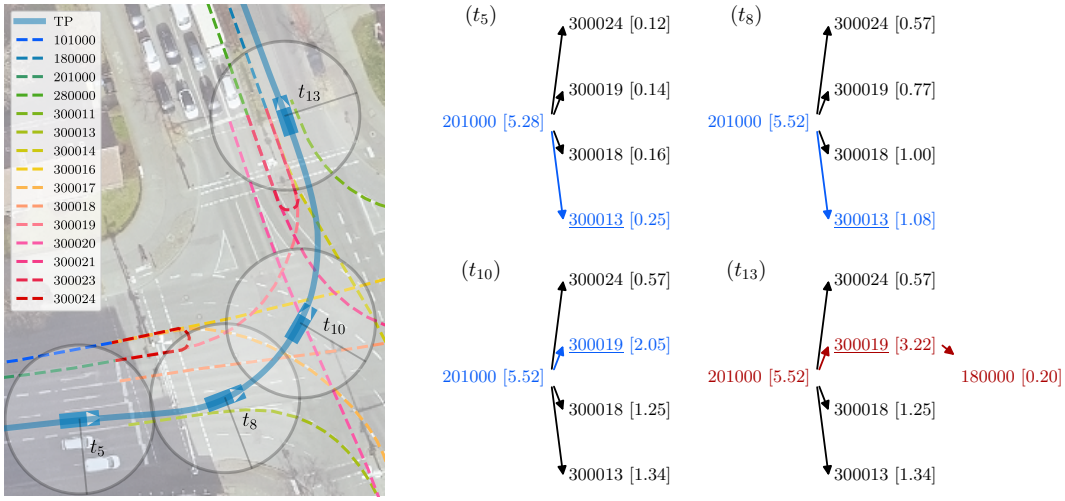


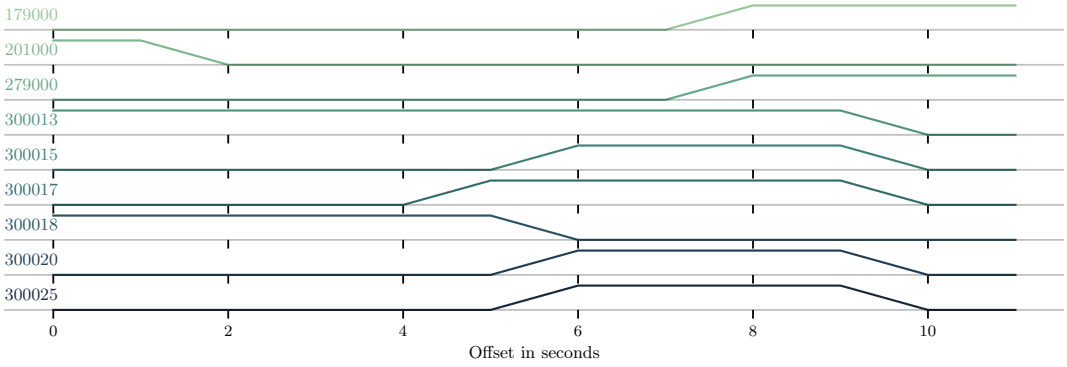
Figure 3.34: Demonstration of the driving-primitive based route estimation approach using a traffic participant performing a `left turn` maneuver on the intersection. *Left:* The trajectory of the traffic participant and the poses for four time instants t_5, t_8, t_{10}, t_{13} with the road search radius illustrated that is used to find the road in the vicinity. *Right:* The evolution of the primitive graph (DAGMaR) for the four time instants, with roads of the intersection represented by their unique identifier and their cumulative mapping likelihood in square brackets. The most likely connected road is underlined, while both, the most likely path during updating DAGMaR and the final path by DAGMaR are highlighted.

The result of the proposed approach uses a DAG of primitives (DAGMaR) for route estimation, which evolution is illustrated in Figure 3.36 for four different times (t_1, t_3, t_7, t_9) analogous to the previous maneuvers. The final path found via DAGMaR as illustrated with the highlighted path in Figure 3.36 at t_9 is $P_{\text{DAGMaR}} = \{201000, 300013, 279000\}$. Similar to the *straight turn* maneuver, all approaches estimate the correct path.

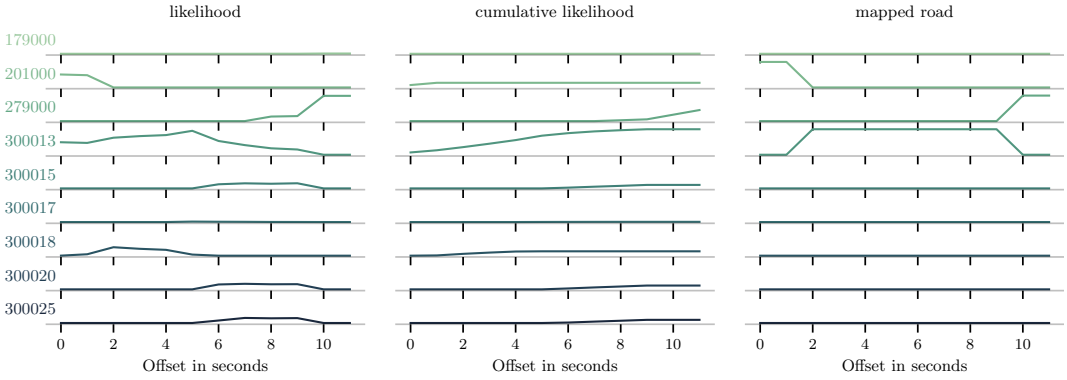
U-turn

The next results we will discuss is for the traffic participant that performs a *u-turn* maneuver. As in the previous section, the roads within the traffic participant's vicinity changes, which is illustrated in Figure 3.37a. In this case, the road user approaches the intersection until t_2 since the vicinity at t_2 is defined by the incoming road 101000 and the outgoing road 201000. Compared to the other use cases, the road user stays within the intersection for a longer period of time from t_3 until t_{20}. It leaves the intersection at t_{21} with a vicinity defined by the same roads (101000, 201000) as while approaching the intersection. Throughout the trajectory, 15 of all 24 roads (62.5%) were within the vicinity at least one time. The correct path of the road user is {201000, 300024, 101000}.

For both baseline approaches and the proposed one, the road mapping likelihood is used to find the taken route of a traffic participant. Figure 3.37b illustrates the mapping like-



(a) The roads in the vicinity of the traffic participant.



(b) The (cumulative) mapping likelihood and the most likely mapped road per time.

Figure 3.35: The road search and mapping results for the traffic participant performing a *right turn* maneuver.

lihood, the cumulative mapping likelihood per road and the most likely mapped road per time and road as a binary signal. The estimated path by MLMP for the *u-turn* maneuver is $P_{\texttt{MLMP}} = \{201000, 300019, 300016, 300012, 101000\}$.

The MLCR approach constructs that path by finding the most likely connecting road. This is illustrated in Figure 3.37b as the cumulative likelihood. For the *u-turn* maneuver, the most likely connecting road is 300019. By using the topology of the intersection, the path is $P_{\texttt{MLCR}} = \{201000, 300019, 279000\}$. Note that this is the same path as for the *left turn* maneuver from above.

The result of the proposed approach uses a DAG of primitives (DAGMaR) for route estimation, which evolution is illustrated in Figure 3.38 for four different times (t_3, t_{10}, t_{15}, t_{20}) analogous to the previous maneuvers. The final path found via DAGMaR as illustrated with the highlighted path in Figure 3.38 at t_{20} is $P_{\texttt{DAGMaR}} = \{201000, 300024, 101000\}$.

For the *u-turn* maneuver, only the proposed approach using DAGMaR finds the correct path of the traffic participant $\{201000, 300024, 101000\}$. The Figure 3.37a shows the reason why MLMP cannot find the correct path. The connecting road for the *u-turn* maneuver 300025 is not within the search area from t_{12} until t_{19}. This is also illustrated in the right

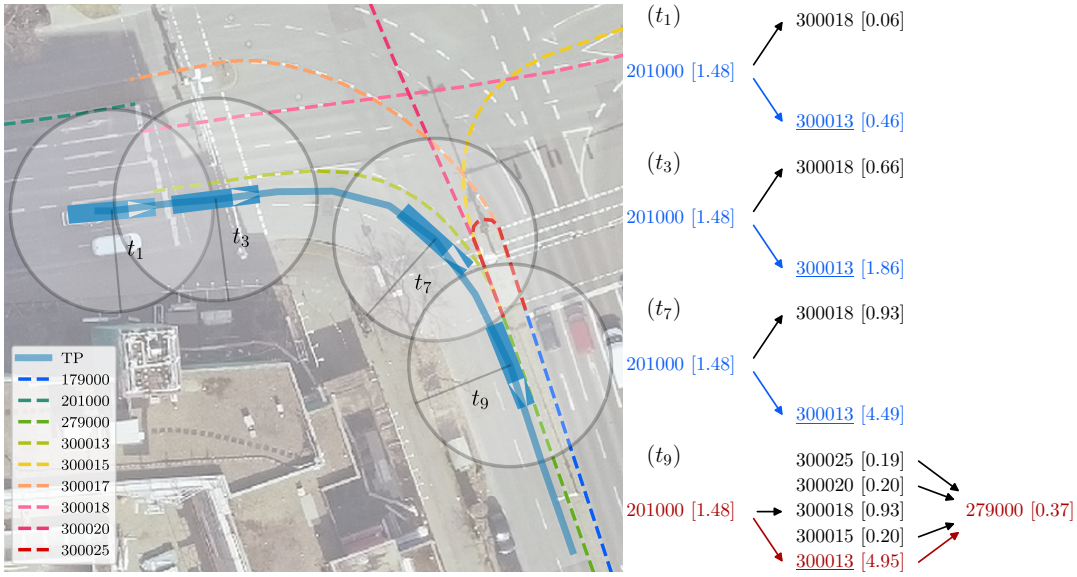
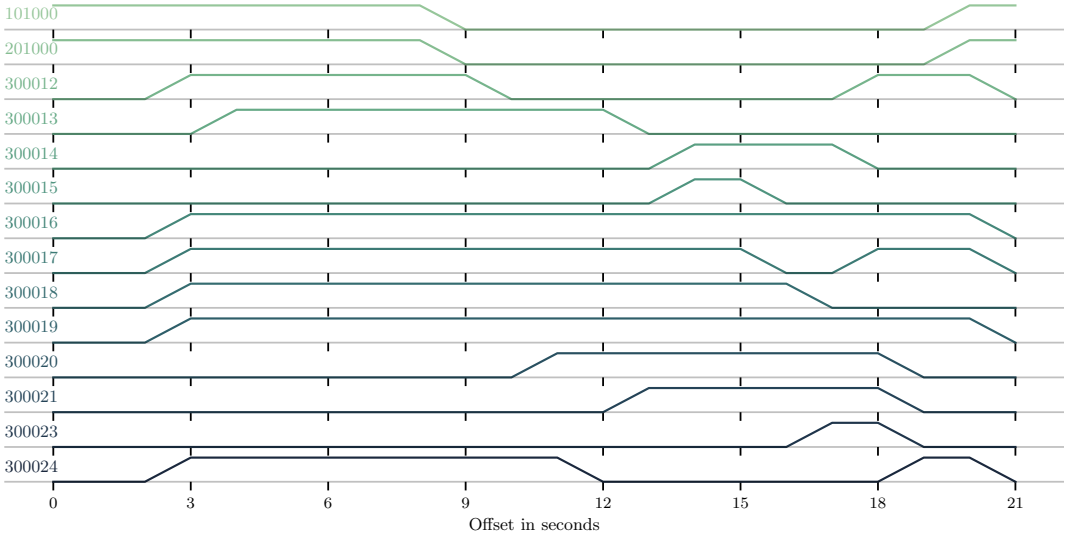


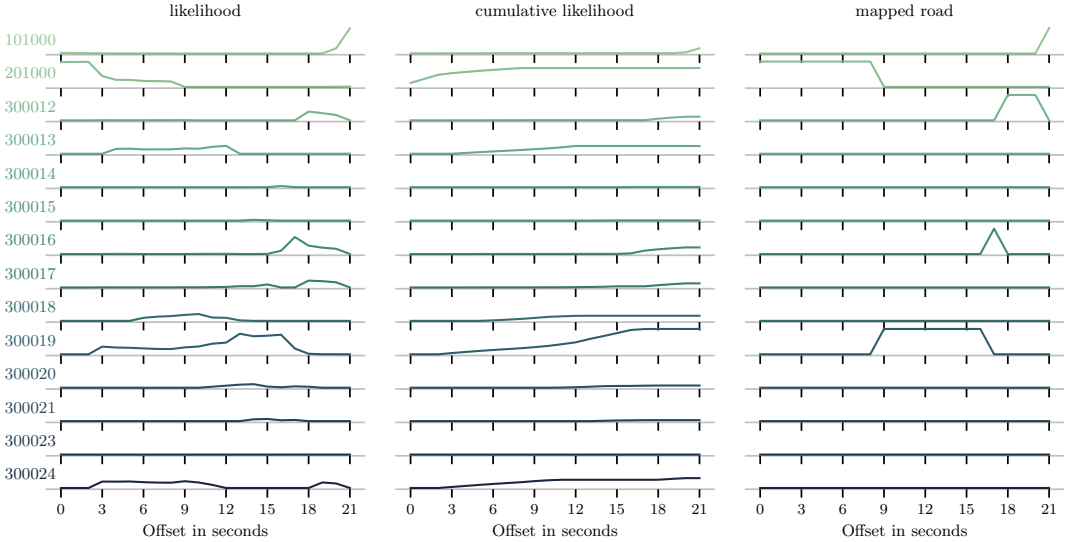
Figure 3.36: Demonstration of the driving-primitive based route estimation approach using a traffic participant performing a `right turn` maneuver on the intersection. *right:* The trajectory of the traffic participant and the poses for four time instants t_1, t_3, t_7, t_9 with the road search radius illustrated that is used to find the road in the vicinity. *Right:* The evolution of the primitive graph (DAGMaR) for the four time instants, with roads of the intersection represented by their unique identifier and their cumulative mapping likelihood in square brackets. The most likely connected road is underlined, while both, the most likely path during updating DAGMaR and the final path by DAGMaR are highlighted.

panel of Figure 3.38, where the road user's search area at t_{15} does not overlap with the connecting road. Consequently, the road user cannot be mapped onto that road. A side effect of this is that the cumulative mapping likelihood of this road will not increase during this time, as shown in Figure 3.37b. This is also the reason why `MLCR` estimates the wrong path. During the previously mentioned period, the road user is close to the road 300019. Because of this, its mapping likelihood increases steadily, so that it becomes the most likely connecting road (see also the right panel of Figure 3.38). The main reason why the proposed approach finds the correct route is based on the aforementioned assumption that a road user enters an intersection via an incoming road and leaves it via an outgoing road. Since the road 300019 is not connected to any outgoing road and the only outgoing road is 101000, which is the successor of 300024, the final path is {201000, 300024, 101000}.

An overall overview of the route estimation results are depicted in Table 3.5, which shows the comparative evaluation of the different route estimation methods. Each row corresponds to a specific method, denoted by labels such as `MLCR`, `MLMP`, `DAGMaR`. The columns represent the different routes, including *straight ahead*, *left turn*, *right turn*, and *u-turn*. For each combination of method and route, a binary notation is used to indicate the accuracy of the estimation. A check mark (✓) signifies that the method successfully estimated



(a) The roads in the vicinity of the traffic participant.



(b) The (cumulative) mapping likelihood and the most likely mapped road per time.

Figure 3.37: The road search and mapping results for the traffic participant performing a *u-turn* maneuver.

the route, while a cross (×) indicates that the method failed to estimate the correct route. The *straight ahead* and *right turn* maneuvers were correctly estimated by all approaches. The *left turn* maneuver by `MLCR` and `DAGMaR`. The *u-turn* maneuver was only identified by `DAGMaR`.

In summary, the baseline approaches lack precision in route estimation if the traffic participant's trajectory does not follow the geometric representation of the roads. This is illustrated in Figure 3.39, which shows the lateral displacement (left panel) and the head-

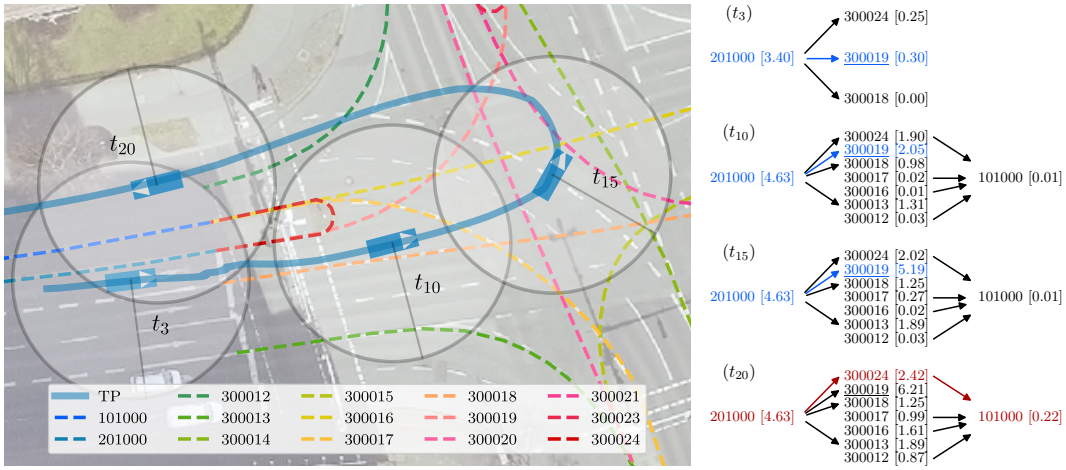


Figure 3.38: Demonstration of the driving-primitive based route estimation approach using a traffic participant performing a *u-turn* maneuver on the intersection. *Left:* The trajectory of the traffic participant and the poses for four time instants t_3, t_{10}, t_{15}, t_{20} with the road search radius illustrated that is used to find the road in the vicinity. *Right:* The evolution of the primitive graph (DAGMaR) for the four time instants, with roads of the intersection represented by their unique identifier and their cumulative mapping likelihood in square brackets. The most likely connected road is underlined, while both, the most likely path during updating DAGMaR and the final path by DAGMaR are highlighted.

ing difference (right panel) of the road user to its mapped pose. In fact, the *straight ahead* and *right turn* road users have only a minor lateral displacement and heading difference to the road 300016 or 300013 respectively. In the *left turn* scenario, the road user has a significantly higher lateral displacement than the *straight ahead* and *right turn* scenarios. Because of this, the road user is mapped to another potentially closer road, which affects `MLMP`. Since `MLCR` searches for the most likely connected road, it is not sensitive to these effects. As long as the road user follows the connecting road and it is in its vicinity, it provides a robust estimation. The *u-turn* scenario is an example where both, `MLMP` and `MLCR` fail due to the high lateral deviation and high orientation difference (see Figure 3.39). Since the road is not within the vicinity of the traffic participant, `MLMP` provides a wrong path. In addition, since the road user follows the path of the road 300019, it becomes the most likely connected road, which is the reason why `MLCR` fails. This does not affect `DAGMaR`. It robustly identifies the routes for all scenarios using the primitive-based approach by considering all road types and the road network's topology to create a DAG of the traffic participant's path through the intersection.

Table 3.5: The route estimation results for the four scenarios and the route estimation approaches.

Method / Maneuver	**straight ahead**	**left turn**	**right turn**	**u-turn**
MLMP	✓	×	✓	×
MLCR	✓	✓	✓	×
DAGMaR	✓	✓	✓	✓

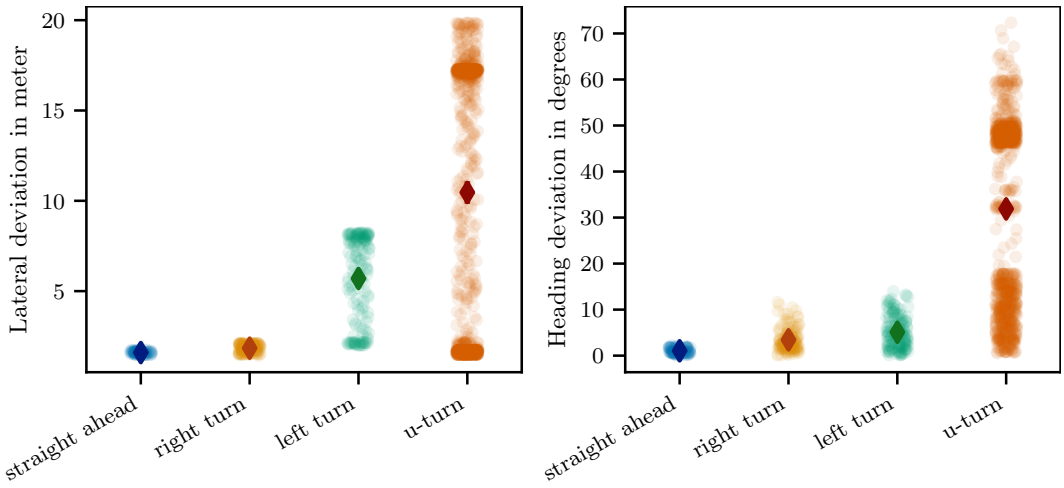


Figure 3.39: Comparison of the geospatial deviation between the road user's trajectory and the road's geometric representation. *Left:* The lateral deviation. *Right:* The heading deviation.

3.5 Summary

This chapter proposed a framework to represent traffic data in terms of primitives based on a concrete use case, by defining a context that is related to the use case and a context-dependent domain of primitives. Several steps are identified to represent driving data in terms of primitives and to model maneuvers based on the primitives.

For a proof-of-concept two case-studies were conducted. The first case study faces the problem of lane-change maneuver identification using the primitive-based representation of traffic data. In particular, lane-changes maneuver are identified and extracted from traffic data collected by the vehicle's on-board sensors. The dataset for evaluation was collected on a German highway as part of the FASva project [89]. In the case study the lane-change was decomposed into more trivial sub-states, which define the primitive domain. For the transformation of traffic data into the primitive domain, the case study shows that a data-driven approach utilizing a Hidden Markov Model (HMM) is a robust method. Moreover, the case study shows that Dynamic Time Warping (DTW) is a reasonable method for the final maneuver classification using the a priori defined maneuver signatures. In fact,

the experiments have shown a reasonable maneuver classification performance with a f1-score of 0.988, although the extracted lane-changes deviate from the manually annotated lane-changes with a mean *f1-score* for maneuver extraction of 0.405.

In the second case study, the framework is applied for the use case of route estimation on an urban intersection. The digital representation of the road network and the trajectory of traffic participants are used to define the primitive domain and context. That is, the roads of the road network are the primitives and the context is defined by the Euclidean distance and heading deviation between the vehicle and its projection on the road. For the transformation of the trajectory into the domain of primitives, a method is proposed that bases on map matching. That is, the traffic data is represented as a DAG of primitives, which allows to use graph algorithms for finding the actual route for crossing the intersection. The experiments using trajectories of traffic participants from the DLR AIM Research intersection have shown that the framework can be applied successfully for the use case of route estimation on an urban intersection.

In summary, the case-studies have shown that the primitive based approach for traffic data representation allows to describe data on a semantical level. Moreover, given a use case or scenario definition, the proposed framework is a useful guideline to systematically identify the relevant maneuvers. A cornerstone of the overall approach is the decomposition of relevant attributes of the traffic data description into primitives domains. The transformation of traffic data into these primitives domains enables to describe maneuvers on a semantical level, which is grounded by the primitives. The first case study in particular shows the whole process, starting with a rather theoretical discussion about a lane-change to the actual application with real-world driving data.

Chapter 4

Maneuver identification using a digital road network representation

This this work uses an hierarchical view on traffic data with scenarios, maneuvers and primitives, as described in Section 2.1.3. In the last chapter, a framework was proposed and demonstrated with case-studies that enables the representation of traffic data with primitives. Furthermore, a methodology was presented to define and identify maneuvers based on these primitives.

In fact, maneuvers are typically identified in real-world traffic data w.r.t the trajectories of objects [97], [98]. That is, the current state (and the history) is analyzed to reason about the maneuver that is being conducted. A drawback of this approach is, however, that trajectories might vary heavily dependent on the environment. For instance, trajectories of vehicles performing a left-turn maneuver may share the same origin and destination. But, the way objects reach their destination and thus how they cross the intersection can be significantly different. Algorithms for maneuver identification need to be able to cope with this variability of trajectories.

This chapter presents an alternative approach to identifying maneuvers of the infrastructure layer by Hartjen *et al.* from Figure 2.9, in particular maneuvers performed by traffic participants to cross an intersection, such as left turn, right turn and straight ahead. This approach utilizes the topological structure and geometrical representation of a road network in OpenDRIVE format. The roads of that digital map serve as generalized reference trajectories and each reference trajectory of an intersection is automatically annotated with a specific maneuver. Due to this, the most likely maneuver conducted by a traffic participant to cross an intersection can be inferred from the reference trajectory to which it is associated, rather than from the trajectory itself. Furthermore, this approach enables the prediction of the most likely maneuver based on the current state of an object due to the *a priori* knowledge of each road's maneuver. In Section 3.4, a method was proposed for solving the route finding problem using our basic building blocks, the primitives. This chapter will therefore focus on the automated maneuver annotation of reference trajectories based on a OpenDRIVE representation of a road network. The efficacy of the methodology is evaluated with established clustering methods k-means, GMM and Hierarchical Agglomerative Clustering (HAC) and the inner-city road network of Brunswick [74].

4.1 Road network partitioning

Since the aim is to identify and annotate maneuvers, the focus is on parts of the digital map where different roads meet each other, such as intersections. The reason for this is that intersections typically connect roads coming from or leading to different directions. Hence, traffic participants need to conduct certain maneuvers to either follow or change the direction towards their final destination. For this purpose, a road network is partitioned into *intersections* and *connections*. Roads of type *connection* connect, as the name suggests, intersections in a road network. This is exemplarily depicted for the inner-city ring of Brunswick in Figure 4.1. The roads interconnecting intersections are depicted in blue whereas intersections are color encoded.

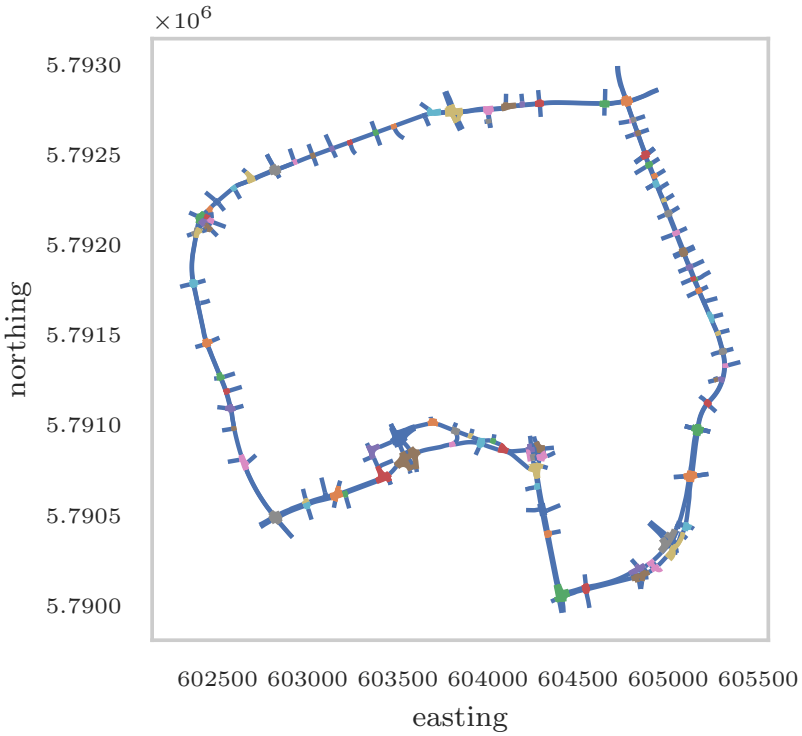


Figure 4.1: The inner-city ring of Brunswick as part of the DLR AIM research facility represented with the reference lanes from the OpenDRIVE format. The intersections in the map are highlighted whereas roads connecting intersections are depicted in blue.

As already mentioned, the majority of infrastructure maneuvers in the urban area take place on intersections or junctions. Hence, all the interconnecting roads are neglected and the focus is on the roads of intersections. As mentioned in Section 2.3 intersections are explicitly designed in OpenDRIVE, each having a unique identifier. Thus, the extraction of roads belonging to an intersection is straight forward.

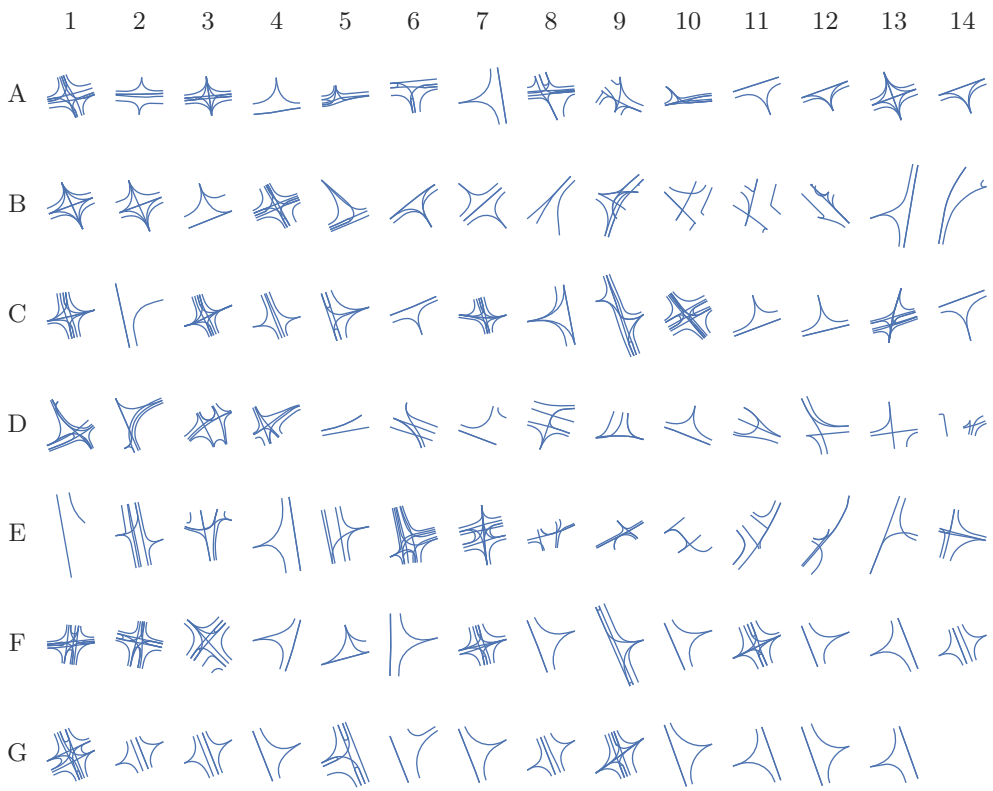


Figure 4.2: All intersections extracted from inner-city ring of Brunswick [74]. Each road is depicted by the reference line from its OpenDRIVE representation. The intersections vary in the number of roads, their shapes and the combination of roads defining the structure of the intersection.

In Figure 4.2 all intersections ($n = 97$) that are present in the inner-city ring are depicted with their *reference line*. The structure of the intersections vary heavily, from simple junctions as in $(B, 14)$[1] or $(D, 5)$ over T-shaped intersections in $(A, 4)$ up to complex intersections as in $(A, 1)$. An automatic mechanism for maneuver annotation needs to support this diversity.

This work proposes a generic data-driven approach by employing unsupervised learning for automatic road annotation. For that purpose, a transformation to each road is applied that enables to represent the heading direction of each road independently of their position on the intersection. This approach allows to collect features from different intersections that can finally be employed to annotate roads with a maneuver.

[1]The nomenclature (R, C) is used to specify a cell in the figure's grid with C as the column and R as the row.

4.2 Maneuvers on urban intersections

For the annotation of roads in the network with infrastructure maneuvers, the "divide and conquer" paradigm can also be applied by dividing a complex space, the overall road network, into more trivial sub states, the maneuvers. In the following, this decomposition of a road network into roads with annotated maneuvers will be briefly formulated.

In general, vehicles either cross an intersection by driving straight through it or they change their direction. Due to this, this work assumes that an intersection consists of multiple roads and the set of roads can be partitioned into two subsets. The first step consists of *straight roads* and the second set of *turning roads*. As already visible in Figure 4.2 the variability in the turning roads is quite high reflecting different turning maneuvers. Due to this, the set will be divided into three subsets denoting the turning maneuvers: *left*, *right* and *u-turns*. Hence, this work assumes that traffic participants can either perform a straight ahead maneuver, left turn, right turn or u-turn on intersections.

Formally defined, let $\mathcal{W}_\mathcal{N} = \{r_1, r_2, \ldots, r_N\}$ be a set of N roads representing a road network $\mathcal{N}$. That network has subsets each of which represents a certain maneuver, *i. e.*, a *semantic network*. At first, let the road network $\mathcal{N}$ has two disjoint proper subsets $\mathcal{W}_\mathcal{N} = \{\mathcal{A}, \mathcal{B}\}$ with $\mathcal{A}$ denoting roads that are part of intersections and $\mathcal{B}$ the roads connecting intersections.

An intersection $\mathcal{I}$ of a network $\mathcal{N}$ can thus be represented as a set of l roads with $l \leq N$, assuming that the network contains at least a single intersection. The two subsets $\mathcal{C}, \mathcal{D}$ of $\mathcal{A}$ are two disjoint sets, $\mathcal{C} \cap \mathcal{D} = \emptyset$, with each of them being a proper subset of $\mathcal{A}$, $\mathcal{C} \subsetneq \mathcal{A}, \mathcal{D} \subsetneq \mathcal{A}$. The first set $\mathcal{C}$ consists of roads driving straight through an intersection and the second set $\mathcal{D}$ of turning roads. To represent the three different variants of turning maneuver (left, right and u-turn), the set $\mathcal{D}$ is divided into subsets $\mathcal{E} \subseteq \mathcal{D}, \mathcal{F} \subseteq \mathcal{D}, \mathcal{G} \subseteq \mathcal{D}$ which are all disjoint $(\mathcal{D} \cap \mathcal{E} \cap \mathcal{F}) = \emptyset$.

Since not all turning variants might exist for an intersection, the subsets $\mathcal{E}, \mathcal{F}, \mathcal{G}$ are no proper subsets of $\mathcal{D}$. Otherwise, an intersection $I_1 = \{r \mid r \in \mathcal{B} \vee r \in \mathcal{E}\} = \{\mathcal{C}, \mathcal{E}\}$ that consists of only straight and left turning roads would not be valid since

$$\begin{aligned} I_1 &= \{\mathcal{C}, \mathcal{D}\} \\ &= \{\mathcal{C}, \{E, \mathcal{F}, \mathcal{G}\}\} \\ &= \{\mathcal{C}, \{E, \emptyset, \emptyset\}\} \\ &= \{\mathcal{C}, \mathcal{E}\} \end{aligned}$$

and thus $\mathcal{E} = \mathcal{D}$ for I_1. Hence, an intersection $I \in \mathcal{N}$ of a semantic network $\mathcal{N}$ can be partitioned into the subsets $\{\mathcal{C}, \{E, \mathcal{F}, \mathcal{G}\}\}$ as depicted in Figure 4.3 with $\mathcal{C}$ denoting straight lines, $\mathcal{E}$ left turns, $\mathcal{F}$ right turns and $\mathcal{G}$ u-turns.

The problem is to unambiguously associate a road r to the correct subset of a network $\mathcal{N}$. That is, the aim is to find its correct semantic meaning in terms of the maneuver that a road user will conduct while crossing an intersection on that road. For that purpose, features needs to be defined that allow to describe the different maneuver unambiguously.

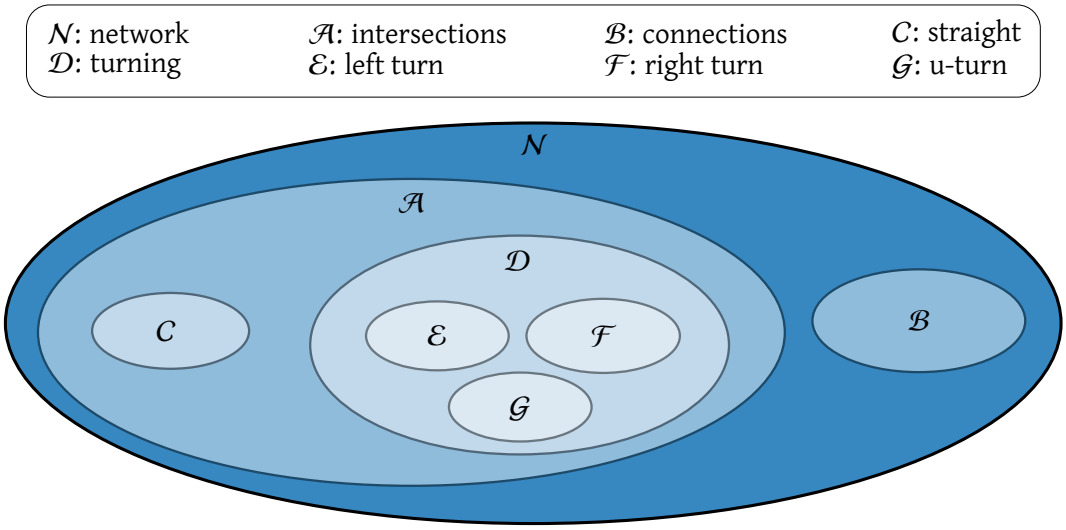


Figure 4.3: The partitioning of roads into maneuver types depicted as a Euler diagram. All subsets in a set are disjoint since a road should be uniquely associated with a maneuver.

4.3 Feature selection for maneuver identification

For the automatic annotation of roads with maneuvers, appropriate features are required that allow to separate the different maneuver types. Since the defined set of relevant maneuvers state the relative heading direction of an object crossing an intersection, *e.g.*, left turn and right turn, the heading of a road should, in turn, allow to infer the road's type.

Let's assume that a road can be represented as a polyline. For roads in OpenDRIVE format, this is a road's reference line. If it is assumed that $\theta \in [-\pi, \pi)$ denotes the orientation difference of a road's polyline to a straight line, it is easy to verify that turning and non-turning roads differ in the magnitude of θ. For instance, straight lanes tend to, obviously, be approximately zero, i.e., $\theta\,(r, C) \approx 0$ and turning lanes have a significant higher absolute magnitude, i.e., $\theta\,(|r|, \mathcal{D}) \gg 0$. This is also depicted in Figure 4.4 showing the intersection $(\mathcal{B}, 7)$ which consists of roads $\{r_0, r_1, r_2, r_3, r_4\}$ with different shapes.

That feature also enables us to separate left and right turning lanes from u-turns by assuming that the absolute magnitude between those two classes differ significantly. What is left is the separation of left and right turns. Apparently, the sign of θ already encodes the heading of a turning lane, i.e., $\theta\,(r, \mathcal{E}) < 0, \theta\,(r, \mathcal{F}) > 0$. In the following, these characteristics are used to cluster the intersections according to the maneuver types. But for that purpose, θ needs to be estimated from a road. If we assume that each road represents a specific heading direction or its change, and thus reflects a specific maneuver, we can further assume that this characteristic change is present by comparing the road's heading at the start and end point. That is, for each road r the unit vectors $\boldsymbol{u}, \boldsymbol{v}$ at the start $\boldsymbol{u}$

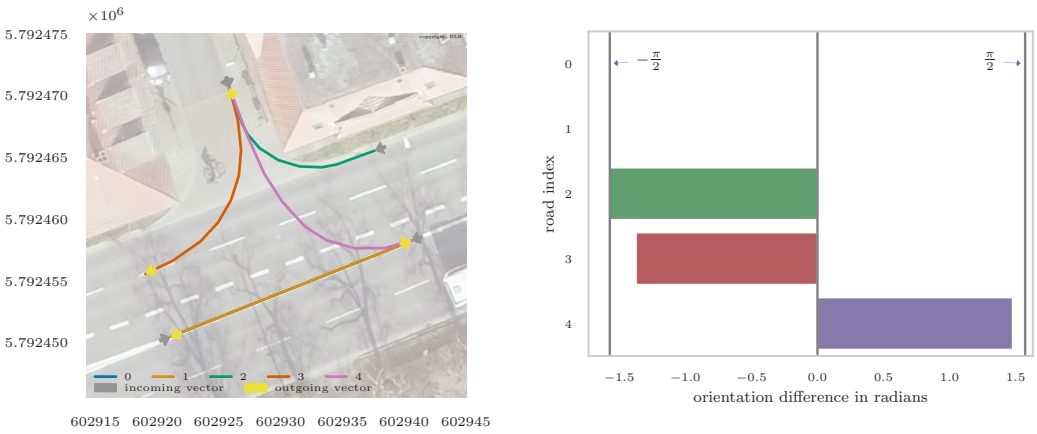


Figure 4.4: The example intersection $(\mathcal{B}, 3)$ of the Brunswick inner-city road network. *Left:* A satellite image with the *reference lines* for each road of the intersection as overlay. At each road, the *incoming* and *outgoing* vectors are shown. *Right:* The orientation difference between the *incoming* and *outgoing* unit vectors for each road is shown denoting the shape of the road. This allows to reason about the maneuver, traffic participants conduct on them.

and end $\boldsymbol{v}$ are estimated, i.e. the *incoming* and *outgoing* vectors of each road as depicted in Figure 4.4 (left) with the yellow and gray arrows for the intersection $(\mathcal{B}, 3)$).

If roads are specified with parameterized geometric models, the unit vector of the tangent in those positions can be derived. Since this work employs a line segment based approximation, the unit vectors at the start and end of each road equals to the unit vectors of the first and last line segment. The signed angle θ between $\boldsymbol{u} = [u_1, u_2]^T$ and $\boldsymbol{v} = [v_1, v_2]^T$ is now defined as

$$\theta = \arctan2\left(\det\left([\boldsymbol{uv}]\right), \boldsymbol{uv}^T\right), \theta \in [-\pi, \pi] \tag{4.1}$$

with `det` giving the determinant of the matrix $[\boldsymbol{uv}]$ and $\boldsymbol{uv}^T$ as the dot product of $\boldsymbol{u}$ and $\boldsymbol{v}$. Note that for unsigned angles, the `arccos` of $\boldsymbol{v}$ projected onto $\boldsymbol{u}$ could be used. But, since the sign of the angle is required to separate left and right turns, the `arctan2` function is employed yielding the required sign. The orientation difference θ is estimated for all roads of intersection $(B, 3)$ according to (4.1) and depicted in Figure 4.4.

Clearly visible are the three different maneuver types. The two roads with a orientation difference close to zero are roads r_0, r_1 depicted as straight lines in Figure 4.4. The two right turning roads r_2, r_3 have a negative orientation difference and the only left turning road r_4 has a positive orientation difference. Note that there is no lane on the intersection performing a *u-turn* so that the semantic network $I_{(B,3)} = \{C, \{\mathcal{E}, \mathcal{F}, \mathcal{G}\}\}$ for this intersection is $I_{(B,3)} = \{\{r_0, r_1\}, \{\{r_4\}, \{r_2, r_3\}, \{\emptyset\}\}\}$.

4.4 Clustering intersections for semantic annotation

The fact that intersections may not contain roads for all maneuver types and the requirement of unique association of a road to a specific cluster or maneuver have implications on the set of methods that can be employed for clustering the digital maps into maneuver categories. There exists different categories of clustering methods that could be employed for this task (see [99] for an overview). Since the aim is to use our *a priori knowledge* about the number of maneuvers and thus clusters, and to verify our methodology, the focus is on *partitioning* and *hierarchical* methods, which allow to define the number of final clusters.

Partitioning algorithms will divide the set of roads into k groups with k stating the number of clusters. For that purpose, a sample is represented by an attribute and a certain objective function is optimized so that samples in a cluster are similar and different clusters are dissimilar — according to the attribute and object function. [99, p. 401] Thus, a road in the example intersection in Figure 4.4 might be falsely annotated with a *u-turn* since no lane of that type is available. This type of algorithm would, however, fit to the requirement of assigning each road to exactly one cluster [99, p. 398].

An alternative would be to employ *hierarchical methods* for clustering trying to build up a tree of clusters [100]. In Figure 4.5 the classes of maneuvers are depicted as a dendrogram. The aim is to either find the level in the dendrogram that matches the number of clusters [100, p. 6] or to define single or multiple thresholds serving as termination criterion [101].

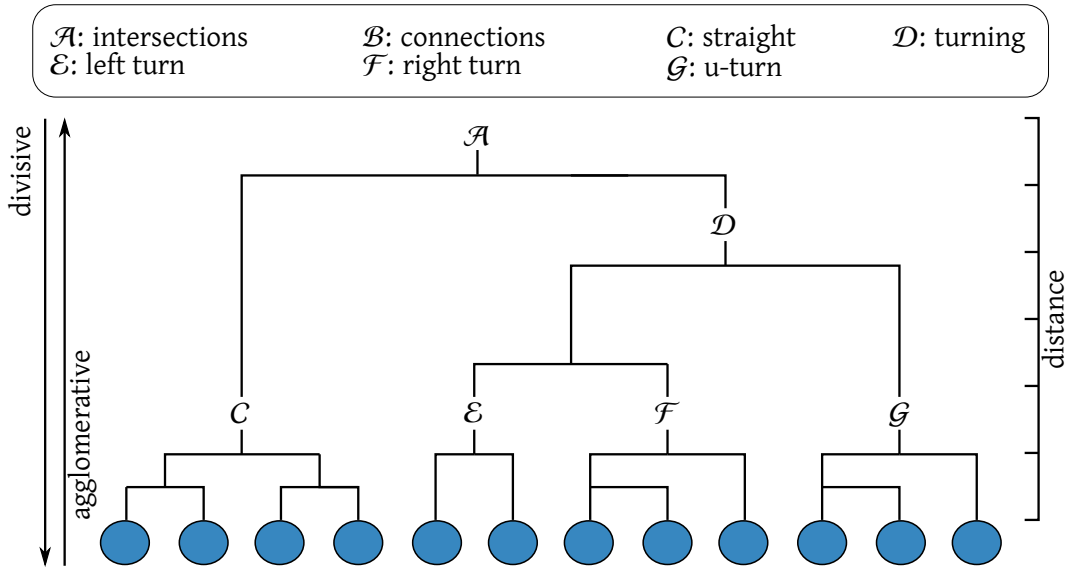


Figure 4.5: An example dendrogram showing the clustering of roads into the maneuver types based on a certain distance metric. This is achieved either by a bottom-up (*agglomerative*) or top-down (*decisive*) approach. A road is represented as blue circle. The dendrogram contains intermediate clusters which are neglected.

In general, both categories of methods could be used for clustering intersection if it could be ensured that for each maneuver category there is at least one road in the road

network. For the example intersection in Figure 4.4, however, this is not the case. The result is that a road might be falsely annotated with a *u-turn* since no lane of that type is available.

A solution for this problem could be to cluster multiple instead of single intersections. If it is assumed that the road network is sufficiently large, it can be further assumed that for each maneuver type at least one road exists. Due to this, one could use both *partitioning* and *hierarchical algorithms* defining only the number of maneuvers or clusters. To verify if this holds for the set of intersections of the Brunswick inner-city road network, the histogram of the orientation deviation for each road is estimated based on (4.1). The distribution is depicted in Figure 4.6 showing five clusters, whereas the first and last cluster belong together. They were split because of the jump in the value range from $-\pi$ to π. This can be addressed by plotting the distribution in polar coordinates as depicted in the top right plot of Figure 4.6 showing the four clusters with a small peak at π. Hence, the assumption holds for this road network, because $\forall r \in \{C, E, F, G\}_{\mathcal{N}} : |r| > 0$, with $\{C, E, F, G\}$ as the subsets of the semantic network $\mathcal{N}$ denoting the maneuver types.

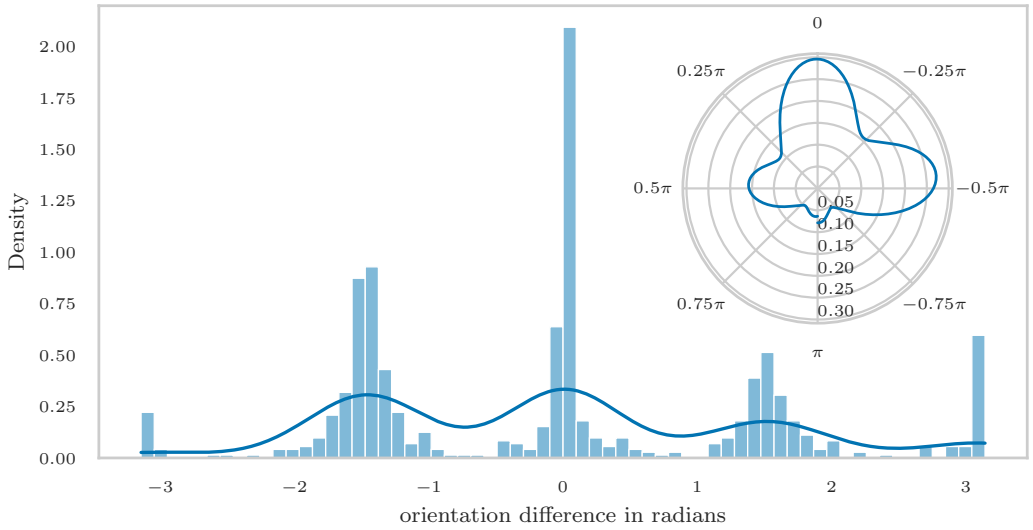


Figure 4.6: The signed orientation deviation distribution for all intersection roads in the Brunswick inner-city road network. There are five clusters visible, whereas the first and last cluster belong together. This is depicted in the top right figure showing the distribution and four clusters in polar coordinates.

Although there exists several *partitioning* and *hierarchical* methods for clustering in the literature, the focus will be on Gaussian Mixture Model (GMM) and k-means as example methods for the first group and Hierarchical Agglomerative Clustering (HAC) for the latter since they are established methods and broadly used in different domains.

Since the GMM and HAC have parameters except the number of components, the performance of the methods will be evaluated for different variants. However, since these clustering methods employ certain distance metrics to evaluate if samples are alike or

not, the jump in the orientation difference depicted in Figure 4.6 would negatively impact the clustering performance. This is due to the fact that $\theta \in [-\pi, \pi)$ and the distance (here the absolute distance)

$$d(\theta_1, \theta_2) = |\theta_2 - \theta_1| \tag{4.2}$$

between two points is not symmetric in every case, *i.e.*, $d(\theta_1, \theta_2)$ is not the *minimum* distance between θ_1, θ_2 for all $(\theta_1, \theta_2) \in [-\pi, \pi) \times [-\pi, \pi)$.

For instance, let r_1, r_2 be both roads of type *u-turn* and having a orientation of, e.g, $\theta_1 = -0.8\pi$ and $\theta_2 = 0.8\pi$. The absolute distance between both points as defined in (4.2) is $d(\theta_1, \theta_2) = 1.6\pi$ although from the top right plot of Figure 4.6 it is evident that the *minimum* orientation difference is $d(\theta_1, \theta_2)_{\text{min}} = 0.4\pi$. Since the distance function may not be manually defined for all used methods, as in our case with the *sklearn* [102] library in Python, two different approaches will be discussed to face this issue. The first uses the vector representation of the orientation difference and clusters the road network in one step. The second approach utlizes the absolute orientation difference and clusters the road network in two steps.

4.4.1 One-step clustering

The first version will cluster the road network into the four classes in a *one step* and recovers the correct class name, the semantic, afterwards. The latter is required since the employed clustering methods may label the clusters different in consecutive runs. The distance estimation problem is tackled by transforming the feature θ. That is, for each road $r \in \mathcal{N}$ represented with its orientation difference θ_r, the unit vector defined as

$$\boldsymbol{u}_r = \begin{bmatrix} \cos\theta_r \\ \sin\theta_r \end{bmatrix} \tag{4.3}$$

is used as feature, with θ_r as the angle between the unit vector $\boldsymbol{u}_r$ and the vector $[1, 0]^T$ in $\mathbb{R}^2$. This transformation allows to employ standard distance metrics for clustering such as the euclidean distance. The distribution after feature transformation is depicted in Figure 4.7 as a scatterplot on a unit circle since all vectors are normalized. Due to the overlapping of points on the unit circle, the distribution computed with a Gaussian Kernel Density Estimator is depicted in the inner-circle showing the four clusters in the dataset.

The results after clustering need to be post-processed to recover the correct semantic, the maneuver type, of each cluster. For that purpose, the assumption about the orientation difference for the maneuvers stated in the beginning is utilized. Let $\hat{\mathcal{N}}$ denote that semantic network after clustering. Furthermore, let's define a set of normalized vectors $S = \{\boldsymbol{s}_1, \boldsymbol{s}_2, \boldsymbol{s}_3, \boldsymbol{s}_4\}$, *i.e.*, $\forall s \in S : \|s\| = 1$ from the same domain as our feature vectors. The vector $s_i \in S$ represents the ith maneuver type in the correct semantic network $\{C, E, F, G\}_{\mathcal{N}}$. The aim is to find a mapping $M : \hat{\mathcal{N}} \xrightarrow{S} \mathcal{N}$ from the subsets $X \in \hat{\mathcal{N}}$ to those in $\mathcal{N}$ using the vectors S.

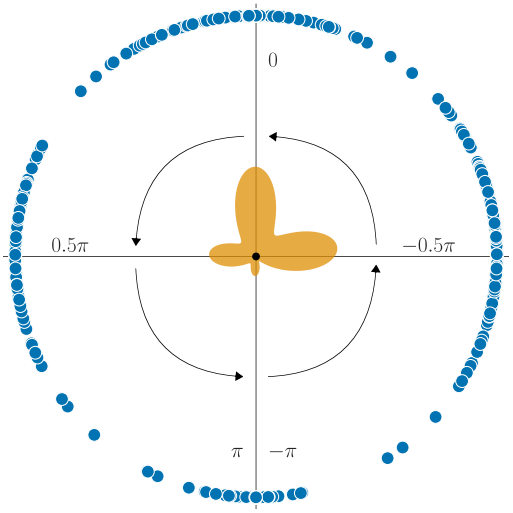


Figure 4.7: The orientation deviation distribution for all roads that are part of intersections in the inner-city network of Brunswick. The feature of each road is depicted as point on the unit circle (in blue), whereas the distribution is depicted in orange computed with a Gaussian Kernel Density Estimator.

Let $\xi(\boldsymbol{s}, r)$ denote the *similarity* between s and the road $r \in X$ with the property of increasing similarity if $\xi(\boldsymbol{s}, r)$ increases, then i is given by

$$i = \underset{j=1}{\overset{|S|}{\operatorname{argmax}}} \sum_{r \in X} \xi(\boldsymbol{s}_j, r) \tag{4.4}$$

and thus the index of the most similar vector, *i.e.*, the vector with the overall highest *similarity score*. Since a road r can be represented as its orientation deviation in the unit circle via (4.3), $\xi(s, r)$ can be defined as

$$\xi(\boldsymbol{s}, r) = \boldsymbol{u}_r \cdot \boldsymbol{s}^T \tag{4.5}$$

and the dot product between $\boldsymbol{u}_r$ and $\boldsymbol{s}$. This holds the property of increasing similarity with increasing $\xi(\boldsymbol{s}, r)$ since the dot product between two unit vectors heading in the same direction becomes one and minus one for two vectors heading in opposite directions. Hence (4.4) becomes

$$i = \underset{j=1}{\overset{|S|}{\operatorname{argmax}}} \sum_{r \in X} \boldsymbol{u}_r \cdot \boldsymbol{s}_j^T \tag{4.6}$$

and returns the index of the most likely maneuver in $\mathcal{N}$ for a certain set X.

For that purpose, however, the vectors in S need to define that represent the maneuvers. Since the orientation deviation of a road as defined in (4.1) is within $(-\pi, \pi]$ and using our assumption about the shape of the roads associated with the maneuvers, the domain is split up into four equally-spaced regions with the vectors $S = \left\{[1, 0]^T, [-1, 0]^T, [1, 0]^T, [-1, 0]^T\right\}$ which are also visible in Figure 4.7 as the four grid lines.

4.4.2 Two-step clustering

The second variant proposed in this work performs the clustering in *two steps*, whereas the second cluster step is performed in the semantic-recover task. The distance problem described before is tackled by taking the halve instead of the full circle. That is, because of $\theta \in [0, \pi]$ the minimum distance is equivalent with the absolute distance between θ_1, θ_2, *i. e.*, $d(\theta_1, \theta_2) = d(\theta_1, \theta_2)_{\min}$. The distribution of absolute orientation difference for all roads that are part of intersections is depicted in Figure 4.8. Due to this transformation, there are only three clusters visible. That is, by taking only the halve of the original domain range, *left* and *right turns* are assumed to be sampled from the same probability density function and only differ in the heading direction.

As with the *one-step* clustering version, the correct class semantic needs to be recovered after clustering. This is straight-forward in this case because the maneuver we can be directly inferred from the sets by their mean or *expected* value. Let $\hat{\mathcal{N}} = \{C, H, G\} = \{C, \{E, F\}, G\}$ denote that semantic network after clustering with its sub-clusters C, H, G for the maneuvers *straight ahead*, *left/right turn* and *u-turn*. The expected value for each set $X \in \hat{N}$ can be estimated with

$$E(X) = \frac{1}{|X|} \sum_{i=1}^{|X|} \theta_i \tag{4.7}$$

so that a set U can be defined with

$$U = \left\{ E(X_1), E(X_2), \dots E(X_{|\hat{\mathcal{N}}|}) \right\} \tag{4.8}$$

of expected values using (4.7) for each subset in $\mathcal{N}$. That is employed to find the correct index $j \in \{1, \dots, n\}$ of X in the final network $\mathcal{N}$ with the correct labels. This is solved by L = argsort U which returns a list of indices $L = \{j_1, \dots, j_n\}$ that contains the location in U if U is sorted in ascending order, *i. e.*, the index of the correct subset in $\mathcal{N}$ and thus the maneuver. The second step of this approach is to split up the set of *left/right turn* maneuvers. As already pointed out, the sign if θ allows to infer the correct subset of H by

$$\begin{aligned} E &= \{r | r \in H \wedge \theta_r > 0\} \\ F &= \{r | r \in H \wedge \theta_r < 0\} \end{aligned} \tag{4.9}$$

so that E contains all *left turn* and F all *right turn* roads.

4.5 Evaluation and discussion

The aim of this approach is to annotate each road of a network according to the maneuver a traffic participant conducts if associated with that road. Furthermore, the results of this approach are used in the following as a basis to derive further maneuver types relating traffic participants to each other via their associated road. Due to this, the performance of this methodology is evaluated.

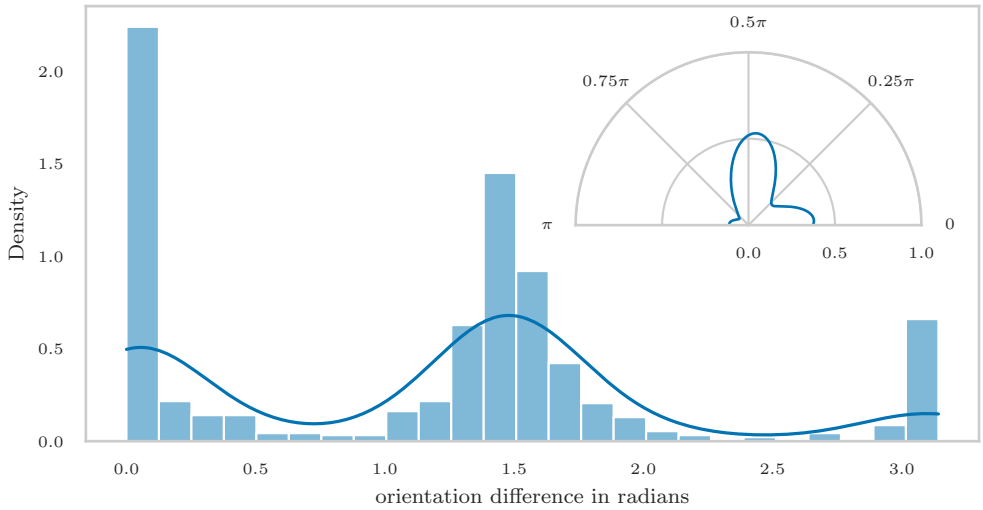


Figure 4.8: The absolute orientation deviation distribution for all intersection roads in the Brunswick inner-city road network. There are three clusters visible: *straights*, *left/right turns* in one cluster and *u-turns*. This is also depicted in the top right figure showing the distribution and three clusters in polar coordinates.

4.5.1 Dataset

For the evaluation of the methodology, the intersections of the Brunswick inner-city road network as depicted in Figure 4.2 are employed. Since it needs to be verified if an annotation by the methodology described in Section 4.4 is correct, all roads were manually labeled. That was conducted by a person that is not associated to this study to ensure that the annotation results are objective. Since this work uses a data-driven approach for maneuver classification, that person should annotate the road with a maneuver according to the change of the driving direction. For labeling, the satellite image of each intersection with an overlay of the road network as in the left image of Figure 4.4 was used.

The number of roads per maneuver and the distribution in the unit circle are depicted in Figure 4.9 for the annotation results of the set of intersections on the Brunswick inner-ring circle. There is no uniform distribution of road classes but *right turns* and *straight* roads are more frequent than *left turn* and *u-turn* roads. In fact, there are 75 *u-turns* which are 10% and 150 *left turn* roads and thus 20.4% of the total number of roads.

4.5.2 Metrics

For qualitative evaluation of the proposed approaches, several metrics exists that allow an objective comparison between results. In the following, the problem is discussed as a multi-label classification problem instead of a typical clustering process since the clustering performance for all maneuver types should be evaluated. That is, the *precision* for each

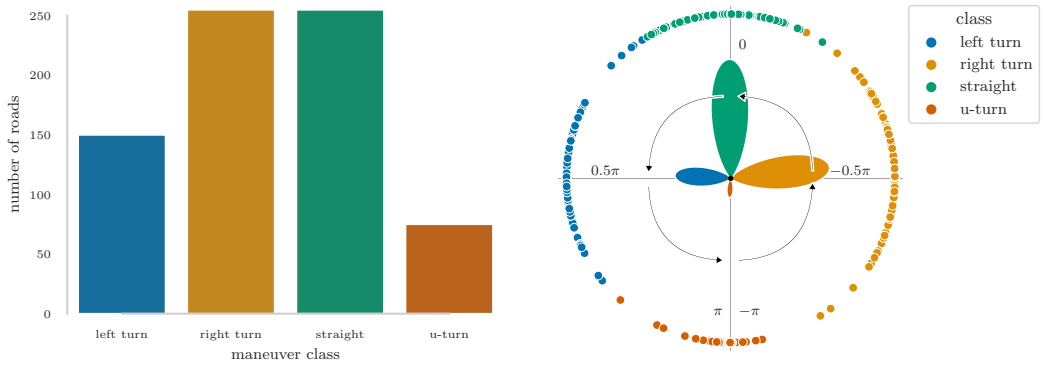


Figure 4.9: The ground truth information of road maneuver types. *Left:* The number of roads per maneuver type. *Right:* The distribution of orientation deviation on the unit circle with the maneuver classes denoted in different colors.

employed clustering method is estimated. If t is the number of correctly annotated roads and $|N|$ is the total number of roads, the precision P is defined as

$$P = \frac{t}{|N|} \tag{4.10}$$

giving the ratio between correctly identified annotated roads to the total number of roads in the network. That is, a value of $P = 1$ means that the all roads are correctly annotated and $P = 0$ that no road is correctly annotated.

For a more in-depth analysis of possible misclassified roads, the *precision* defined in 4.10 can, however, not be used since it just provides a single value about the overall precision. Instead, the *confusion matrix* is employed since it allows a relation between the number of annotated roads by a method to the ground truth reference for each maneuver type. But, in order to put emphasis on the *difference* from the correct classification, a Confusion Difference Matrix (CDM) is used. A perfect method would generate a Confusion Difference Matrix (CDM) of only zeros since the number of annotated roads match the correct ones. The *diagonal* of a CDM highlights the overall difference in the number of roads for that maneuver. A negative value states that roads are missing in this set and a positive value that too many roads are annotated. Each *row* in a CDM depicts the difference from the ground truth related to the maneuver type of that row. Due to this, the sum of each row is zero. A *column* gives information about the number of roads that are annotated by the method to that particular maneuver type and thus highlights the *sensitivity* for that particular maneuver. Hence, the CDM allows to infer how the roads are incorrectly annotated by a method for each maneuver.

Table 4.1: The clustering precision of the *one-step* version for k-means, GMM with different covariance types and HAC with different linkage types. The precision is given as the ratio of correctly annotated roads to the total number of roads for each maneuver type. The GMM with spherical covariances performs best in general.

		Maneuver				
Method	Variant	Straight	Left turn	Right turn	U-turn	Mean precision
k-means		1.000	0.973	0.984	1.000	0.989 ± 0.011
GMM	diag	1.000	0.973	0.984	1.000	0.989 ± 0.011
	full	1.000	0.973	0.988	1.000	0.990 ± 0.011
	spherical	0.996	1.000	0.988	0.987	**0.993 ± 0.006**
	tied	1.000	0.980	0.988	0.987	0.989 ± 0.007
HAC	average	1.000	0.947	0.984	1.000	0.983 ± 0.022
	complete	1.000	0.960	0.992	0.987	0.985 ± 0.015
	single	1.000	0.960	-	0.987	0.737 ± 0.124
	ward	1.000	0.960	0.992	0.987	0.985 ± 0.015

4.5.3 Results

As already pointed out in the beginning of this section, the GMM, k-means and HAC is employed for clustering. The number of final clusters is dependent on the used approach with $n = 4$ for the *one-step* version and $n = 3$ for the *two-step* version. For the GMM different degree of freedoms of the covariance matrix are evaluated. For the HAC different linkage variants are evaluated since they significantly affect the clustering process.

The results for the *one-step* approach are depicted in Table 4.1 showing the precision defined in (4.10) of all methods for each maneuver type and the mean precision for each method. Overall, the GMM with spherical covariance matrix achieves the best performance with a mean precision of 0.993 ± 0.006. For the HAC method, the *complete* and *ward* linkage variant performs best with a precision of 0.985 ± 0.015. The *simple* method was not able to identify *right* turns correctly leading to a mean precision of 0.737 ± 0.124. k-means lacks precision in identifying *left turns* with a precision of 0.973 and thus achieves a mean precision of 0.989 ± 0.011.

The results for the *two-step* approach are denoted in Table 4.2 also showing the precision of all methods for each maneuver type and the mean precision defined in (4.10) for each method. In this case, the GMM with spherical covariance matrix also achieves the best performance as in the *one-step* approach with a slightly better mean precision of 0.995 ± 0.005 instead of 0.993 ± 0.006 for the *one step* variant. For the HAC method, the *average* and *ward* linkage variant now performs best but with a slightly higher precision of 0.989 ± 0.011 instead of 0.986 ± 0.015 for the *one-step* method. The *simple* linkage variant is to not able

Table 4.2: The clustering precision of the *two-step* version for *k*-means, GMM with different covariance types and HAC with different linkage types. The precision is given as the ratio of correctly annotated roads to the total number of roads for each maneuver type. The GMM with spherical covariances performs best in general.

		Maneuver				
Method	Variant	Straight	Left turn	Right turn	U-turn	Mean precision
k-means		1.000	0.973	0.980	1.000	0.988 ± 0.012
GMM	diag	1.000	0.980	0.984	1.000	0.991 ± 0.009
	full	1.000	0.980	0.984	1.000	0.991 ± 0.009
	spherical	0.996	1.000	0.996	0.987	**0.995 ± 0.005**
	tied	1.000	0.987	0.988	0.987	0.990 ± 0.006
HAC	average	1.000	0.973	0.984	1.000	0.989 ± 0.011
	complete	1.000	0.713	0.816	1.000	0.882 ± 0.123
	single	1.000	-	-	0.933	0.483 ± 0.281
	ward	1.000	0.973	0.984	1.000	0.989 ± 0.011

to identify *left turn* and *right turn* maneuver leading to a mean precision of 0.483 ± 0.281. *k*-means achieves a mean precision of 0.988 ± 0.012 over all maneuver types.

To further analyze the performance of the approaches for each maneuver type, the confusion matrices are estimated for all methods, except the *single* variant of *HAC* because of its low performance, and both variants (see Figure 4.10).

A perfect method would generate a CDM of only zeros since the number of annotated roads match the correct ones. The *diagonal* of a CDM highlights the overall difference in the number of roads for that maneuver. A negative value states that roads are missing in this set and a positive value that too many roads are annotated. Each *row* in a CDM depicts the difference from the ground truth related to the maneuver type of that row. Due to this, the sum of each row is zero. A *column* gives information about the number of roads that are annotated by the method to that particular maneuver type and thus highlights the *sensitivity* for that particular maneuver. Hence, the CDM allows to infer how the roads are incorrectly annotated by a method for each maneuver.

The confusion matrix of *k*-means illustrated in Figure 4.11a, for instance, contains the numbers $\{2, 0, -5, 3\}$ for the roads that are part of the maneuver type F in the third row. For this specific case, five roads are missing in the set F. Two roads are falsely associated with C and three roads with G. The diagonal contains the values $\{0, -4, -5, 0\}$ stating the number of missing roads in each set, *e.g.*, no missing road for F but four missing roads for G. Hence, *k*-means is able to identify all *straight* and *u-turn* roads — as also depicted in Table 4.1 with a precision of 1.00 for these maneuver. But, it is quite *sensitive* since it annotated three roads as *u-turns* which are actually part of the set F, as denoted in the

column for *G*. Moreover, six roads are classified as *straight* although four roads are of type *left turn* and two of type *right turn* as denoted in the *C* column.

A noticeable difference from the ground truth is, for instance, visible in the confusion matrix of *HAC (complete)* in Figure 4.11a. There are 43 roads of type *E* that are annotated as 4 times *C* and 39 times *G*. Noteworthy is, that the method's performance is significantly better in the *two step* version. Roads of type *E*, *F* are not classified as *u-turns* anymore.

The GMM (spherical) achieves the best overall performance for the *one-step* and *two-step* version with the *one-step* version being a slightly better. For both variants, the method fails to identify one road as *straight* but classifies it as *right turn*. Furthermore, a road of type *u-turn* was classified as *left turn* in both variants. The only difference is in the *right turn* class. In fact, the *two-step* version falsely identified two roads as *u-turns*. In both versions, the method is able to identify all *left turns* as denoted in the row *E* which contains only zeros. Indeed, the method is reasonable accurate in the identification of left turns with no road being falsely associated with a *left turn*.

Another perspective on the clustering results is depicted in Figure 4.11. For all methods (except the *HAC (single)* variant) and both versions, the orientation difference for each road is shown on a unit circle as in Figure 4.7 (right) for the ground truth information. Moreover, the classification result is shown with a correct road denoted as circle and incorrect one as cross. Note that the crosses of incorrectly annotated roads are moved into the circle and the distribution in the inner circle is scaled just for visualization purpose. It is evident from the figures, that methods tend to classify roads that are at the border between two maneuver classes. In fact, the GMM (spherical) distributions for the *one-step* and *two-step* versions clearly show that the only difference is in the annotation of the road in the lower right edge between the sets for *u-turn* and *right turn* roads.

4.5.4 Discussion

The orientation difference of the falsely annotated roads are quite close to the associated class clusters. Question regarding the matching accuracy still remain open. Are the roads really associated to the wrong maneuver classes? Was there an error in manual road annotation that led to this result? A look on the intersection might help to answer those questions. Furthermore, it might also allow to infer extensions to the method that can be faced in future works.

The three roads annotated with the wrong maneuver by the *GMM (Spherical)* method using the *two-step* belong to three different intersections. Each intersection and the roads it consists of are depicted in Figure 4.12 with the falsely classified road in blue. The actual and estimated maneuver type are shown in the title with the first maneuver as the reference and the second as the maneuver type inferred by the method.

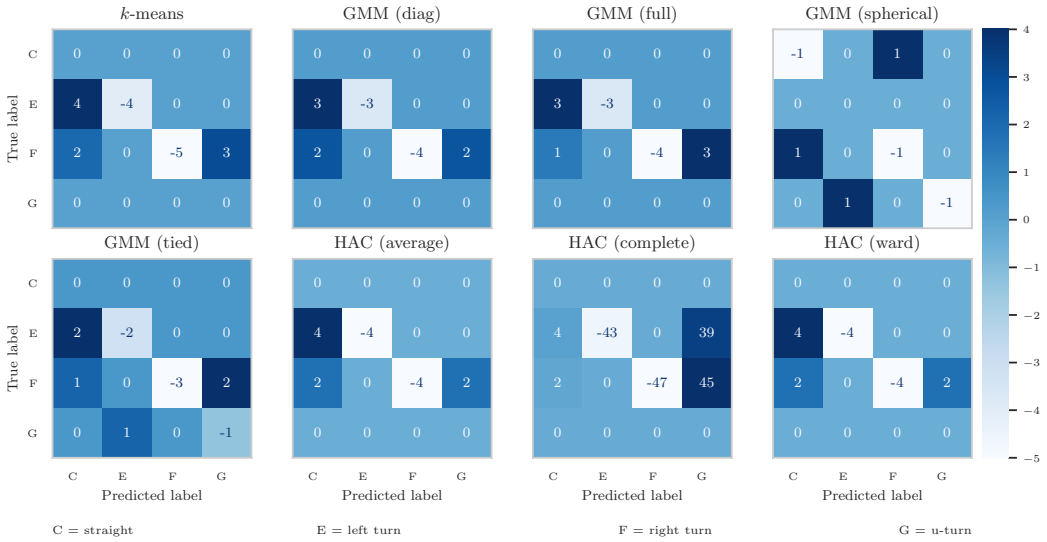
The road in the first intersection was misclassified by the method as *straight* but manually annotated as *right turn*. The road in the second intersection is classified as *right turn* but manually annotated as *straight*. The last intersection contains a road which is annotated as *left turn* although it is manually associated to a *u-turn*. Overall, those roads really seem to be edge-cases. The road in the first intersection is classified as *straight* due to the low orientation difference. But, the annotation of *right turn* is also plausible since a traffic par-

ticipant merges into the road and changes the heading direction to the right. Especially, if not only the line is considered, but also the satellite image of the source road. The road in the second image is another special case since traffic participants cross the intersection on the misclassified road. As such, the road should be of type *straight* as annotated manually. The annotation of the road in the third intersection as *left turn* by the method is reasonable, but in this context a *u-turn* is more correct since the road also contains both lanes that are part of the roundabout. Hence, a vehicle is able to reach its origin. The misclassification might be prevented by splitting up the road into lanes belonging to the roundabout and the other lanes. This allows to distinct between the *u-turn* in the roundabout and a *left turn* for the other lanes.

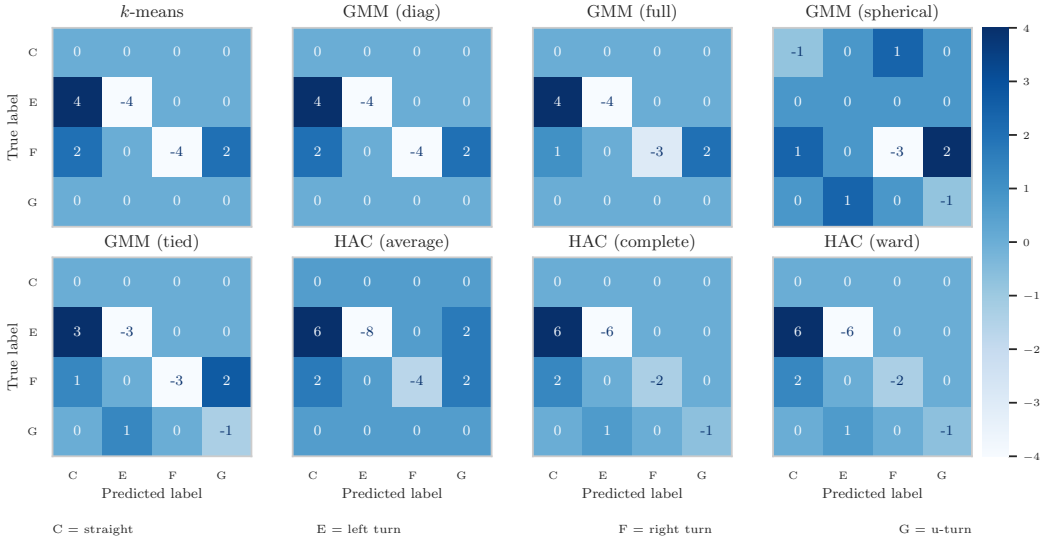
4.6 Summary

This chapter proposes an approach that uses *a priori knowledge* about the set of possible maneuvers on intersections to transform a digital map into a *semantic network*. This follows the overall methodology by dividing a complex environment into more trivial subspaces which are easier and more robust to identify. This *semantic network* allows inferring the maneuver, a traffic participant will conduct, based solely on the road the traffic participant is associated with, thus decoupling the maneuver identification from trajectory analysis, which is prone to error due to the variability of trajectories. In Chapter 4.5 the results of experiments is shown, in which the overall approach is evaluated using a manually labelled digital map of the inner-city ring in Brunswick that is part of the DLR AIM test site.

The information provided by the proposed approach also enables to infer different scenarios. For instance, we might be interested in situations in which a vehicle performs a left turn maneuver, thus crossing other lanes, with road users approaching on these oncoming lanes. This scenario is of interest due to the potential conflict between the traffic participants. In the next chapter, a methodology will be proposed for the identification of such scenarios. That is, the aim is to identify scenarios with multiple road users involved that share the same spatial location of the available traffic infrastructure, thus potentially being in conflict, which is interesting from different point of views, *e. g.*, traffic safety analysis or road user behavior modelling.

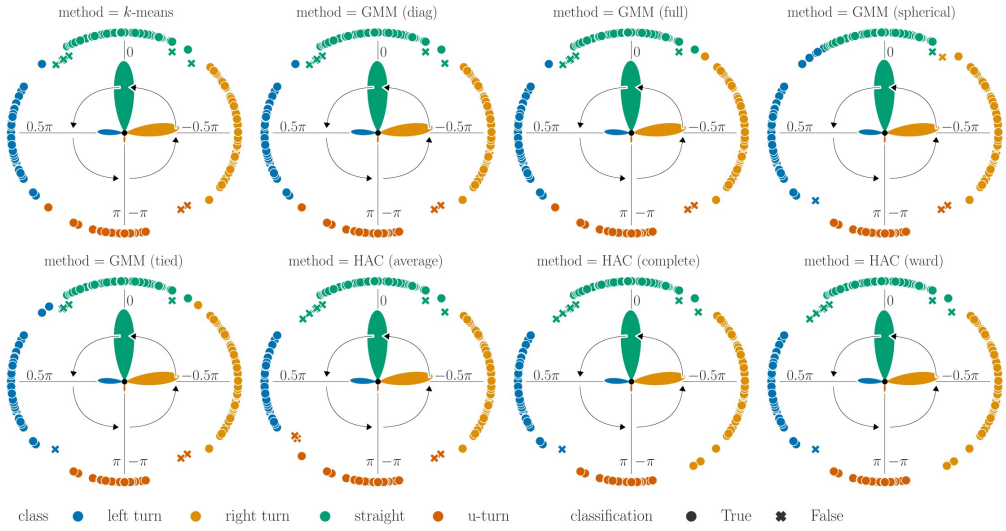


(a) The *one-step* version

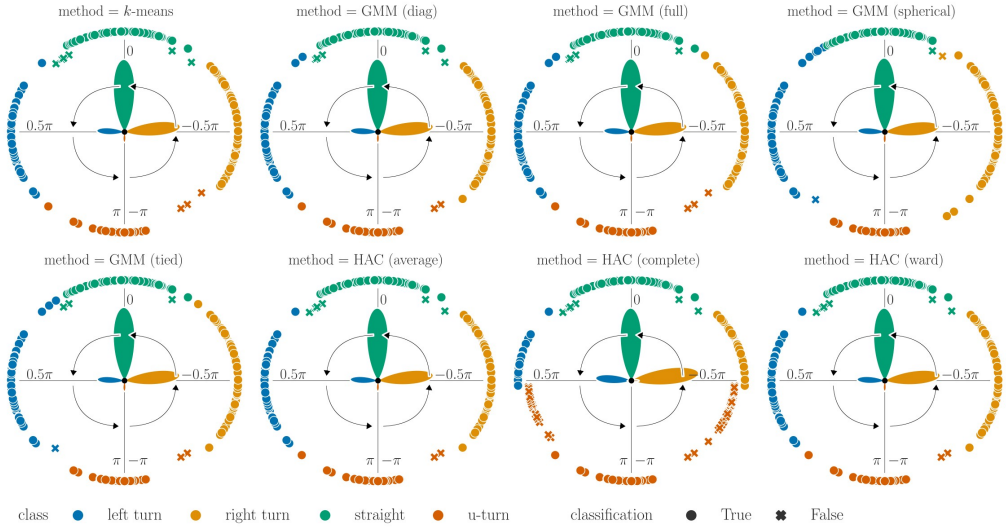


(b) The *two-step* version

Figure 4.10: The Confusion Difference Matrices for all evaluated clustering methods. *(a):* The results of the *one-step* version. *(b):* The results of the *two-step* version. Each CDM highlights the difference to the actual set of maneuvers. It allows to infer the annotated maneuver for a misclassified road.



(a) *one-step* version



(b) *two-step* version

Figure 4.11: The distribution of orientation difference θ for each road in the network on a unit circle after clustering with the *one step* version *(a)* and *two step* version *(b)*. The maneuver types are color encoded, correctly annotated roads are denoted with a circle and falsely annotated roads with a cross. For each maneuver type, the distribution is depicted in the inner-circle scaled for visualization purpose.

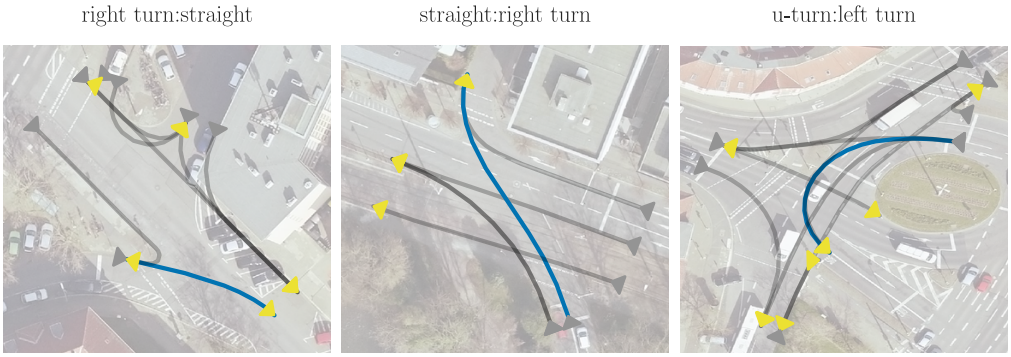


Figure 4.12: The three misclassified roads by the *GMM (Spherical)* method using the *two-step* variant. The title denotes the maneuver type of the reference and the method. The misclassified roads are highlighted in blue.

Chapter 5

A systematic approach for scenario identification

In the previous chapter, maneuvers conducted by road users are inferred from a semantically enriched road network in OpenDRIVE format. In the following, the objective O.1 is addressed by proposing a method to systematically identify scenarios in the urban domain and building a KB of scenarios. The presented method combines a knowledge-based with a data driven approach. The former is utilized to define scenarios and identify scenarios hypothesis. The data-driven approach uses the methods presented thus far in this work and the identified scenario hypothesis to extract concrete scenarios from real-world traffic data for the validation of the scenario hypothesis.

The methodology is demonstrated exemplarily using the DLR AIM Research Intersection in Brunswick. Furthermore, the focus is on the first three variants of space-sharing conflict scenario (SSCS), in the following referred to as *obstructed scenario*, *merging scenario* and *crossing scenario*. In particular, the aim is to identify hypotheses of scenarios solely based on the digital representation of the road network and focus on motorized road users. That is, if all traffic participants are removed from the example situations depicted in Figure 2.11a—2.11e, and only take the road topology and geometry into account, identifying scenarios due to an unconstrained or constrained head-on path is not possible. For their definition, the trajectories of the traffic participants are required. For instance, if the cyclist is removed from the situation depicted in Figure 2.11e, both cars are unlikely to share the same space at the same time in the future. Moreover, the SSCS due to unconstrained head-on paths are unlikely on the intersection depicted in Figure 2.1 since they typically occur where traffic participants can move freely without restrictions. Since this is not the case for the DLR AIM research intersection, we will focus only on the three scenario variants.

5.1 Foundation

Before introducing the proposed methodology for scenario identification and describing the knowledge base, basic terminologies that are relevant for the remainder of the chapter are first described.

5.1.1 Knowledge Base

In the context of computer science, or artificial intelligence in particular, a KB is a component that contains knowledge of a specific domain in terms of so-called *sentences* that are represented in a *knowledge representation language*. [81] These *sentences* are used to describe entities in the world, certain facts about them and relationship among them. They are usually categorized as either being assertional or terminological. In the context of *Description Logic (DL)*, which are a family of knowledge representation languages [103], this categorized led to the definition of a assertional box (ABox) and terminology box (TBox) as the building blocks of a KB. [104][103]

The *sentences* in the **TBox**, or the *terminological axioms*, are used to express concepts, such as `Dog` and `Human` and relationships, such as `hasOwner`, which can be combined to formulate constraints such as $\exists$`hasOwner.Dog` $\subseteq$ `Human` to state that only humans can be owner of dogs.

The axioms of the TBox can be used to define properties of *individuals*. These individuals can be any facts about things in our world. For instance, with `Dog(Edwin),Human(Svenja)` we can state that the individual Edwin belongs to the concept of *Dog* and the individual Svenja to the concept of *Human*. Moreover, with `hasOwner(Edwin,Svenja)` we can state the Svenja is the owner of Edwin. The set of such assertions is the **ABox** and the named individuals within these assertions the ABox individuals. [103]

5.1.2 Ontologies

The knowledge base will contain information about the scenarios and the traffic participants that are involved within and potentially the relationship to their environment. Ontologies are a way to formally define the concepts, relationships and axioms from above such that they are readable by both, human and machines. In fact, they use DL languages as the mathematical foundation for this purpose [105].

The first well-known definition of an Ontology in the context of computer science is from 1993 by Gruber defining an ontology as an "*explicit specification of a conceptualization*" [106], which was extended in 1998 by Studer *et al.* stating that an "*ontology is a formal, explicit specification of a shared conceptualization*" [107]. Emphasizing the use-case of knowledge sharing due to the fact that ontologies are widely adopted in the context of knowledge representation in general and the so-called Semantic Web in particular, where ontologies are "*establishing a common terminology between agents*" [103].

Today, ontologies are usually defined and represented using the Web Ontology Language (OWL), which is a specification of the World Wide Web Consortium (W3C). Ontologies based on the OWL are represented using the RDF format, where statements have the form of (subject S, predicate P, object O) triples and every entity is uniquely identified using an Unique Resource Identifier (URI). A statement in the ontology thus states that the subject S has the property P with value O, with S, O being entities in the ontology that are specified using an URI and P is either a URI or any literal value. [105] Due to this triple-based representation and the unique identification of concepts and relationships within ontologies, they are typically visualized as graphs. [105]

The Figure 5.1 illustrates the representation of ontologies using a graph structure with the example from above. The ontology with the axioms defined above are created with the ontology editor Protégé [108]. Since `Svenja` and `Eddi` are both individuals, there are two corresponding triples in the ontology. This is shown in Figure 5.1 with the connecting arcs between the two nodes and the `owl:NamedIndividual` concept. Furthermore, the relationship that `Svenja` is the owner of `Eddi` is also represented with the arc between the nodes with the attribute `hasOwner`. The constraint that only humans can be owners of dogs is also defined in the ontology by defining the `domain` and `range` of the `hasOwner` relationship as `Dog` and `Human`. Furthermore, the `owl:disjointWith` attribute is used to state that something that is a `Dog` cannot be a `Human`. Note that in the remainder of this work, the Visual Notation for OWL Ontologies (VOWL) [109] is utilized to visualize concepts and relationships among them in an ontology.

One of the strength of ontologies is that knowledge can be extended automatically based on the axioms defined in the KB. In the context of ontologies, this process is called *reasoning* and *reasoner* the corresponding engine that performs the inference. One of this engine is Pellet [110], which is used throughout this work. Revisiting the example depicted in Figure 5.1, let's assume that we have not explicitly stated that `Eddi` is a `Dog` and `Svenja` is a `Human`, *i. e.*, the red arcs are removed. But, since it is stated that `Svenja` is the owner of `Eddi`, that the `domain` is `Dog` and the `range` is `Human` of the `hasOwner` relationship, a reasoner can automatically infer that `Eddi` is a `Dog` and `Svenja` is a `Human`, while also providing a justification for the inference. [110] For instance, the explanation for `Eddi` being a `Dog` consists of the axioms (`Eddi`, `hasOwner`, `Svenja`) and (`hasOwner`, `domain`, `Dog`).

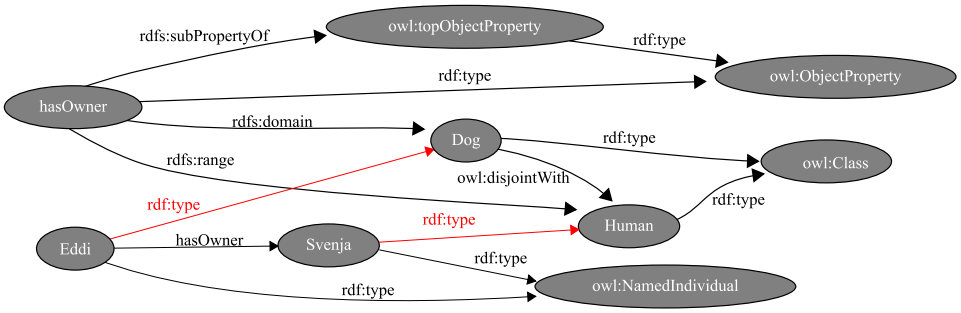


Figure 5.1: Illustration of the example ontology as RDF graph with `Eddi` the `Dog` and `Svenja` the `Human` owner.

5.1.3 SPARQL

Ontologies allow to represent knowledge in human and machine readable format either in single ontologies or distributed across multiples that can be shared and reused in specific applications. While this allows to create sophisticated Knowledge Base (KB), this is only useful if we are able to retrieve information from the KB similar to querying information from a relational database (RDB). Since ontologies in OWL are represented in RDF, the SPARQL can be utilized to query the KB. Similar to the OWL, SPARQL was developed

and standardized by the World Wide Web Consortium (W3C), with its first release version published in 2008.

Statements in SPARQL follow the RDF specification that allows to refer to entities in the ontology using their URI. For instance, using the example KB from above, the SPARQL query in Listing 5.1 can be used find all humans and their dogs. The `SELECT` statement is used to define the variables `?dogOwner` and `?dog` that will appear in the result. The `WHERE` clause allows to match against specific patterns in the RDF graph. Since the `hasOwner` relationship links `Dog` with `Human` individuals, the variables will be used as subject and object.

Listing 5.1: The SPARQL query to search for all human individuals and their dogs.

```
SELECT ?dogOwner ?dog
WHERE {
   ?dog :hasOwner ?dogOwner .
}
```

5.2 Methodology for scenario identification

The concepts and methods presented in the previous chapter provide the foundation to systematically identify scenarios from real-world traffic data. The overall methodology is composed of four tasks (see Figure 5.2). The first step is to extract the OpenDRIVE representation of the intersection of interest, because hypotheses of SSCS are derived from it in the next step. Those hypotheses provide the basis to extract scenario candidates from real-world traffic data, *i. e.* traffic participants that are potentially involved in such a scenario. Finally, the scenario hypotheses are validated with the scenario candidates. The individual steps will be described in the following. But at first, some required definitions are introduced.

5.2.1 Fundamental definitions

Let an urban road network $N = \{W, \phi\}$ be defined by the roads $W = \{w_1, w_2, \dots\}$ it consists of and the connection ϕ between roads, *i. e.*, the road network's connectivity. Then, an intersection I is a subset of the road network $I \subseteq N$. Furthermore, the definition of OpenDRIVE is followed and group the roads of an intersection W_I in three types

$$W_I = \{W_{\text{in}}, W_{\text{connect}}, W_{\text{out}}\} \tag{5.1}$$

where *incoming roads* W_{in} are used by traffic participants to enter the intersection, *outgoing roads* W_{out} to leave the intersection and *intermediate roads* W_{inter} connect incoming with outgoing roads.

The *connectivity* $\phi(w_i, w_j)$ of a road network denotes that a traffic participant can reach a road w_j from another road w_i without visiting any road in between. It is assumed that a road user will always enter and leave the intersection on different roads. The connectivity is a set of tuples

$$\phi = \left\{(w_i, w_j) \in W^2 \mid w_i \neq w_j\right\} \tag{5.2}$$

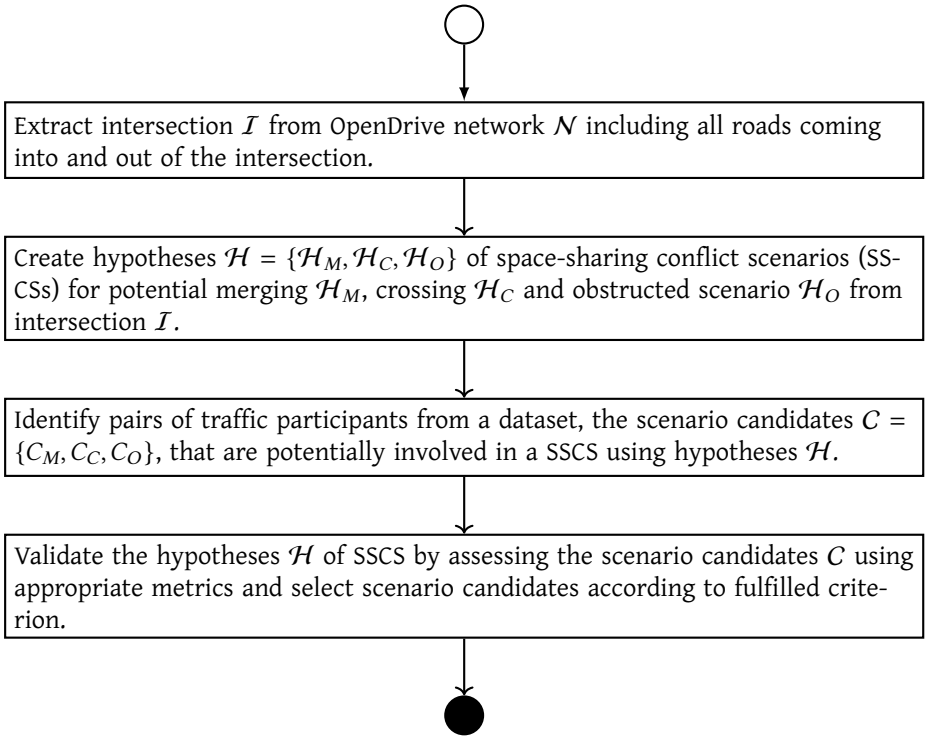


Figure 5.2: The approach for SSCS identification in real-world traffic data using the digital representation of an intersection.

stating that road w_i is connected to w_j. If ϕ_+ and ϕ_- denote the successor and predecessor of a road, w_j is the *successor* $w_i = \phi_+(w_i)$ of w_i and w_j the predecessor $w_j = \phi_-(w)$ of w_i. Note that roads have only one direction, *i.e.*, $\forall(w_i, w_j) \in \phi : (w_j, w_i) \notin \phi$.

5.2.2 Hypothesis definition

The intersection connectivity defined in (5.2) and grouping of roads according to (5.1) allow to define the hypotheses of SSCS. Thus, let $\mathcal{H}_I = \{\mathcal{H}_M, \mathcal{H}_C, \mathcal{H}_O\}$ be the set of hypotheses for the intersection I with $\mathcal{H}_M$ the merging, $\mathcal{H}_C$ crossing and $\mathcal{H}_O$ obstructed scenarios. The sets of hypotheses are defined as follows.

Two traffic participants can become involved in a *merging scenario* if they are associated to two different roads having the same successor and that successor is an outgoing road. Let $w \in W_{\text{out}}$ be an outgoing road of the intersection. The set of merging scenario hypothesis $\mathcal{H}_M$ is thus defined as

$$\mathcal{H}_M = \left\{(u, v) \in W^2 \mid u \neq v, \phi(u, w), \phi(v, w)\right\}. \tag{5.3}$$

Two traffic participants can become involved in an *obstructed scenario* if they are associated to two different roads having the same predecessor and that predecessor is an incoming

or a connecting road. Let $w \in \{W_{\text{in}}, W_{\text{connect}}\}$ be either an incoming or connecting road. Then, the set of obstructed scenario hypothesis $\mathcal{H}_O$ is defined as

$$\mathcal{H}_O = \left\{(u,v) \in W^2 \mid u \neq v, \phi(w,u), \phi(w,v)\right\}. \tag{5.4}$$

Two traffic participants can become involved in a *crossing scenario* if they are associated to two different roads with intersecting geometrical representation. Moreover, both roads have different predecessors and successors. Recap that $\phi_-(w), \phi_+(w)$ are the predecessors and successors of a road w, then the set of crossing scenario hypothesis $\mathcal{H}_C$ is defined as

$$\begin{aligned}\mathcal{H}_C = \Big\{ (u,v) \in W^2 \mid \text{intersect}(u,v),\\ \phi_-(u) \cap \phi_-(v) = \emptyset,\\ \phi_+(u) \cap \phi_+(v) = \emptyset \Big\}\end{aligned} \tag{5.5}$$

with intersect(u,v) denoting that the geometrical representation of the two road's (u,v) intersect.

5.2.3 Hypothesis validation

The set of hypotheses created using the above definitions include road pairs that might not provide real scenarios, i.e., the hypotheses are *invalid*. Since we only use the digital representation of the road network for hypotheses creation, we assume that no traffic control mechanism exist, e.g., traffic rules, traffic signs or traffic lights. In fact, at least traffic rules usually apply limiting the set of hypotheses. Consequently, we will validate the hypotheses of SSCS. If we assume normative behavior of traffic participants, we can utilize the trajectory data of traffic participants for that task. That is, let $\mathcal{X} = \{X_1, X_2, \dots\}$ be a dataset of trajectories, with each trajectory associated to the roads of an intersection. In addition, let C_M, C_O, C_C be sets of traffic participant pairs that are potentially involved in a merging, obstructed or crossing scenario, i.e., the *scenario candidates*. The set $C_i \in \{C_M, C_O, C_C\}$ of all candidates of a SSCS variant is thus defined as

$$C_i = \left\{ (a,b) \in \mathcal{X}^2 \mid a \neq b, f_i(a,b) \right\} \tag{5.6}$$

with $f_i(a,b)$ as a function denoting that two different trajectories (a,b) are potentially involved in the SSCS variant $\mathcal{H}_i$.

The sets of scenario candidates defined in (5.6) and the sets of scenario hypotheses (5.3), (5.4), (5.5) allow to validate the scenario hypotheses and thus giving the space-sharing conflict scenarios of a dataset. That is, a hypothesis $h = (u,v)$ of a scenario variant $\mathcal{H}_i \in \{\mathcal{H}_M, \mathcal{H}_C, \mathcal{H}_O\}$ is valid if

$$\exists (a,b) \in C_i : g_i(a,b,h) \tag{5.7}$$

holds with $g_i(a,b,h)$ as a function denoting that the traffic participants (a,b) of the scenario candidate are indeed involved in the hypothesis h of the scenario variant $\mathcal{H}_i$. Hence,

the trajectories of a dataset $\mathcal{X}$ can be grouped pair-wise to space-sharing conflict scenarios, so that all scenarios of variant $S_i \in \{S_M, S_O, S_C\}$ are given by

$$S_i = \left\{ (a,b) \in C_i \mid \exists h \in \mathcal{H}_i : g_i(a,b,h) \right\} \tag{5.8}$$

and that all road combinations of an intersection $\hat{H}$ providing SSCS are defined as

$$\hat{H} = \left\{ h \in \mathcal{H}_i \mid \exists (a,b) \in S_i \right\} \tag{5.9}$$

which are the *valid scenario hypotheses*.

The definitions above can be divided into four tasks as depicted in Figure 5.2. At first, the intersection of interest is extracted from a road network. This step is optional if there is only one intersection in the network. The intersection's roads are grouped according to (5.1) and the intersection's connectivity as defined in (5.2) is derived. Afterwards follows the creation of SSCS hypotheses according to (5.3), (5.4), (5.5), which are used to identify scenario candidates with a dataset of trajectories according to (5.6). Finally, the hypotheses of SSCS are validated with the scenario candidates. In the following, we will describe each task in more detail.

5.3 Knowledge base of scenarios

In the last section, the overall methodology was described for the identification of SSCS. In the following, the knowledge base of SSCS is introduced and in particular how to represent the definitions of the previous section in the knowledge base. For that purpose, an ontology-based approach using OWL 2 is utilized due to the formal nature of ontologies and SPARQL to find hypotheses of SSCS in an intersection, while validating those hypotheses using *reasoning* on real-world data.

The overall process for this approach is illustrated in Figure 5.3. The knowledge base consists of three so-called TBox entities *Connectivity*, *Hypothesis* and *Participant*, each for an information source or ABox, and the overall set of individuals which will be extended iteratively. For that purpose, facts are deduced using *reasoning* given the ABox, TBox and the results of the previous state (the knowledge base's current state). The ABox `Topology` represents the topological information of an intersection, *i. e.*, how roads of an intersection are connected to each other. The `Geometry` ABox contains the geometric representation of the roads and is used to represent and use the geometrical relationship of roads in the knowledge base. The `Trajectories` ABox contains information about the road users and is based on real-world data. In the following, the three TBox are described and how the definitions of the previous section is represented in the knowledge base.

The *Connectivity* TBox illustrated in Figure 5.4 using the VOWL [109] contains facts about an intersection, its roads and their relationship. In particular, the TBox contains definitions for the different entities and relations shown in the DAG in Figure 5.8 such as the *incoming*, *connecting* and *outgoing* roads, and their relationship via the `has_successor` property. A major benefit of this ontology-based approach is, as already shown using the example in , is that we can utilize reasoning to automatically infer the `has_predecessor`

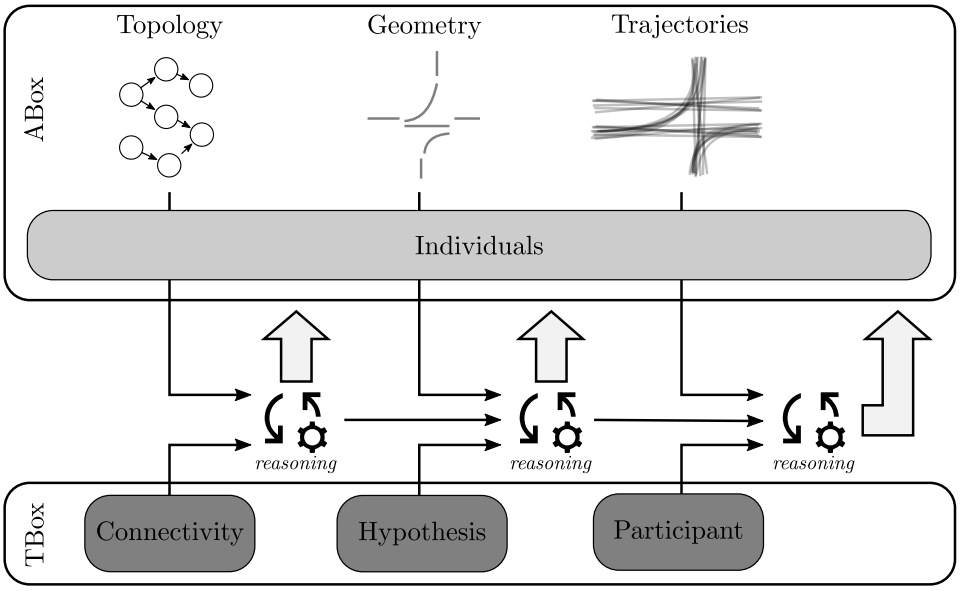


Figure 5.3: The intersection network topological information, road geometries and trajectory data from the real-world are employed to identify SSCS and build the knowledge base.

attribute by stating that it is the inverse of `has_successor`. This is illustrated in Listing 5.2 using the Manchester syntax [111].

Listing 5.2: The `has_predecessor` property is the *inverse* of the `has_successor` property.

```
28 ObjectProperty: has_predecessor
36    InverseOf:
37       has_successor
```

The ontology-based approach also allows defining the different road types of an intersection. For instance, the definition of a `ConnectingRoad` is shown in Listing 5.3 using the relationship between roads and the concrete road type. With the `EquivalentTo` axiom used in Listing 5.3 we state that a `ConnectingRoad` connects an `IncomingRoad` to an `OutgoingRoad` via the properties `has_predecessor` and `has_successor`. After reasoning, a `Road` individual linked to an `IncomingRoad` individual via the `has_predecessor` property and to an `OutgoingRoad` individual via the `has_successor` property will be classified as `ConnectingRoad`. The `Connectivity` TBox also contains similar definitions for the `IncomingRoad` and `OutgoingRoad`.

Listing 5.3: The definition of a `ConnectingRoad` in Manchester syntax.

```
97  Class: ConnectingRoad
98
99     EquivalentTo:
100       Road
101        and (has_predecessor some IncomingRoad)
102        and (has_successor some OutgoingRoad)
```

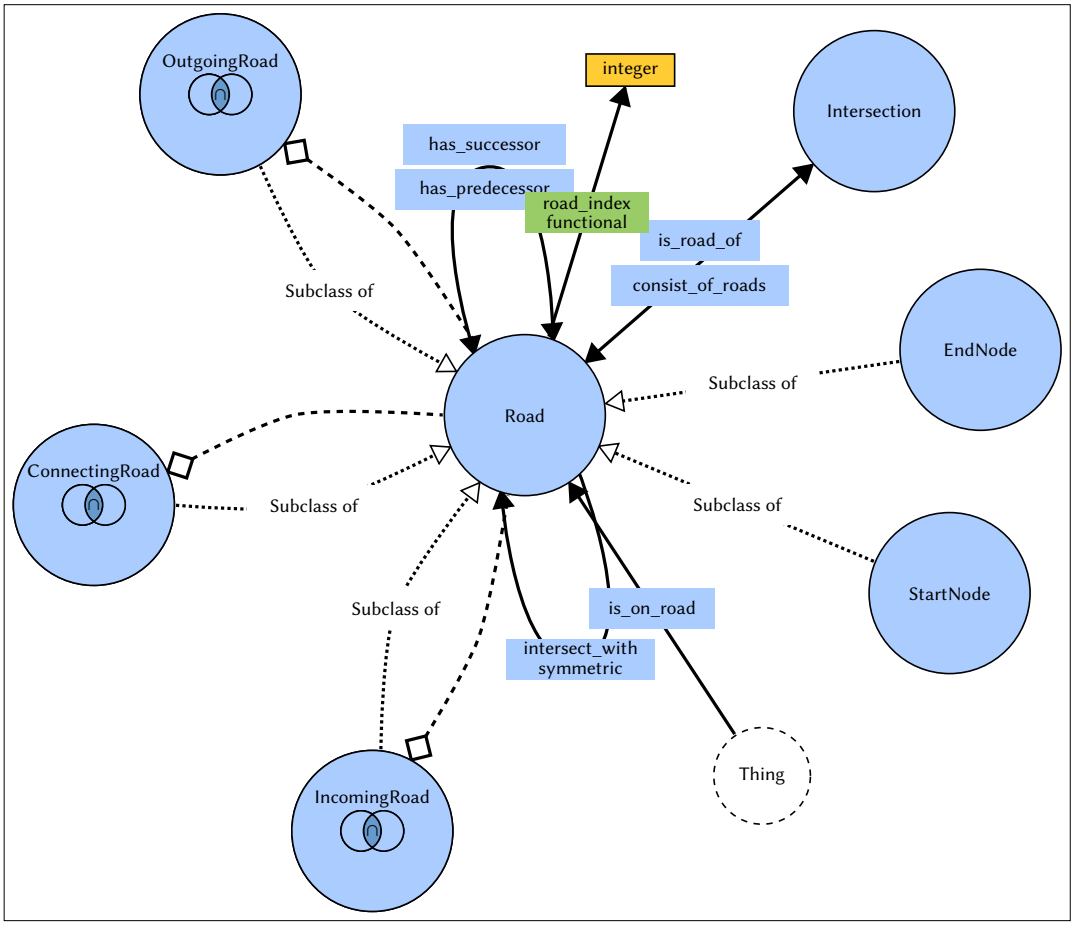


Figure 5.4: The *Connectivity* TBox of the knowledge base visualized in VOWL.

The facts of the `Connectivity` TBox are used in the next step of the approach illustrated in Figure 5.3 to represent the hypotheses of SSCS that are defined in (5.3), (5.5), (5.4). The *Hypothesis* TBox is illustrated in Figure 5.5 showing the class definitions of the three SSCS variants of interest: `CrossingHypothesis`, `MergingHypothesis` and `ObstructedHypothesis`. The three classes are linked to roads of the `Connectivity` TBox via the `consists_of_roads` property. That is, since a SSCS describes a situation with at least two traffic participants involved that are associated to two roads (cf. *e. g.* equation (5.5)), we define that a `Hypothesis` is an entity that consists of two roads via the axiom defined in Listing 5.4. Due to this, every individual in the *ABox* that is linked to two individuals of type `Road` via the `consists_of_roads` property will be classified as a `Hypothesis`.

Listing 5.4: The definition of a `Hypothesis` in Manchester syntax.

```
106 Class: Hypothesis
107
108    EquivalentTo:
```

```
109       consists_of_roads exactly 2 conn:Road
110
111    SubClassOf:
112       owl:Thing
```

Unfortunately, the definition of the three SSCS is not straightforward in an ontology in OWL 2 since OWL is not able to express all relations between individuals. Moreover, individuals need to be stated explicitly in an ABox and cannot be defined by reasoning. We could solve this problem by creating `Hypothesis` individuals of all road pair combinations of an intersection and use Semantic Web Rule Language (SWRL) rules to classify the hypothesis individuals. This approach will, however, populate the KB with irrelevant hypothesis[1]. Another approach to solve the problem, which is employed in the following, is to use SPARQL and search for road pairs in the KB matching the criteria (cf. (5.5), (5.3), (5.4)) and adding the results as hypothesis individuals to the KB. For instance, we create individuals of `CrossingHypotheses` by searching for roads pairs with the SPARQL query shown in Listing 5.5.

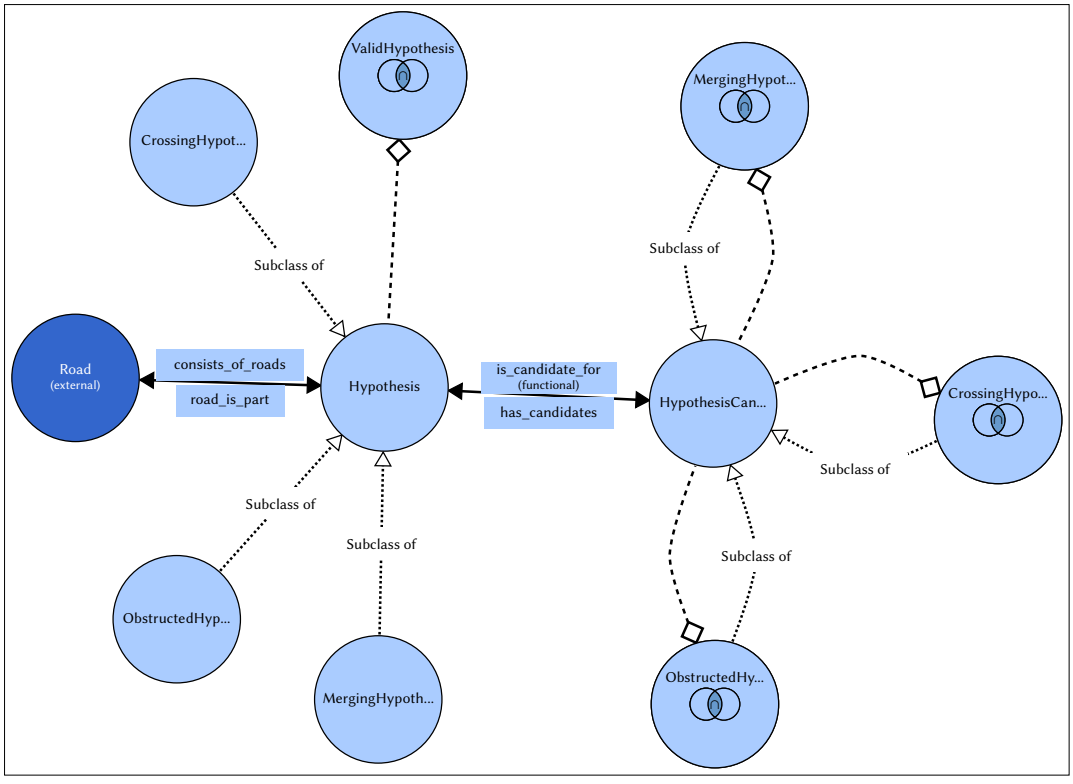


Figure 5.5: The *Hypotheses* TBox of the knowledge base visualized in VOWL.

[1]The hypotheses are irrelevant for the problem faced in this chapter and not in general.

Listing 5.5: The SPARQL query to search for intersecting roads having no common predecessor or successor.

```
1  SELECT ?r1 ?r2 WHERE {
2
3       ?r1 conn:intersect_with ?r2 .
4
5       ?r1 conn:has_successor ?r1_s .
6       ?r2 conn:has_successor ?r2_s .
7
8       ?r1 conn:has_predecessor ?r1_p .
9       ?r2 conn:has_predecessor ?r2_p .
10
11      ?r1 conn:road_index ?idx1 .
12      ?r2 conn:road_index ?idx2 .
13
14      FILTER (str(?idx1) < str(?idx2) && ?r1_s != ?r2_s && ?r1_p != ?r2_p)
15   }
```

In fact, the definition in (5.5) is translated using the properties defined in the *Connectivity* TBox. That is, we search for two roads that *geometrically intersect* with each other in line 3, which is the first condition in (5.5). The second and third conditions are depicted in lines 5-9 by getting the successors and predecessor of both roads in combination with the filter statement in line 14 to ensure that the successors and predecessor are different. The query will return road pairs which are added as individuals of type `CrossingHypothesis` to the knowledge base. To find individuals for the *merging* and *obstructed* scenario, similar SPARQL queries are defined w.r.t (5.3) and (5.4).

As already denoted in Section 5.2 the hypotheses of SSCS need to be validated. In this work, trajectories of traffic participants from the real-world are used for that task. The `Hypothesis` *ABox* contains the `ValidHypothesis` class to indicate that a `Hypothesis` is valid. For the validation, scenario candidates are extracted from the real-world dataset of road user trajectories by associating them to the different roads of the intersection. The `Hypothesis` ABox contains the `HypothesisCandidate` for that purpose, with each of the three SSCS variants represented by a separate subclass. The property `has_candidates` establishes the connection between a `Hypothesis` and its candidates. This allows defining the SSCS subclasses as a `HypothesisCandidate`. For instance, the `CrossingHypothesis-Candidate` can be defined as shown in Listing 5.6. That is, a candidate will be classified as a `CrossingHypothesisCandidate` if it is associated to a `CrossingHypothesis` via the `is_candidate_for` property.

Listing 5.6: The definition of a `CrossingHypothesisCanddiate` in Manchester syntax.

```
96  Class: CrossingHypothesisCandidate
97
98     EquivalentTo:
99        HypothesisCandidate
100       and (is_candidate_for some CrossingHypothesis)
```

The relation between a `HypothesisCandidate` and a `Hypothesis` not only allows defining the SSCS subclasses, but also classifying a `Hypothesis` as a `ValidHypothesis` according to (5.9) as shown in Listing 5.7. That is, a `Hypothesis` becomes valid and thus classified as `ValidHypothesis` if there are any individuals of `HypothesisCandidate` associated to it.

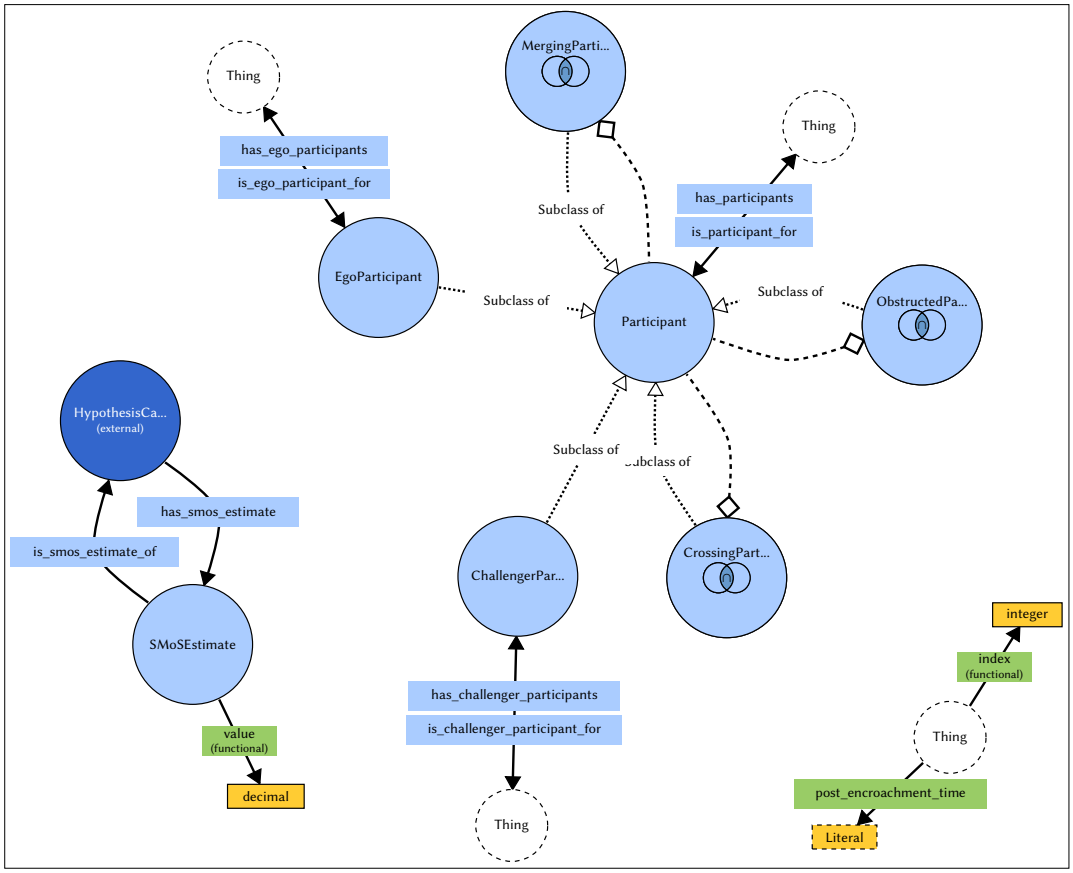


Figure 5.6: The `Participant` *TBox* of the knowledge base.

Listing 5.7: The definition of a `ValidHypothesis` as a `Hypothesis` with candidates in Manchester syntax.

```
153 Class: ValidHypothesis
154
155    EquivalentTo:
156       Hypothesis
157        and (has_candidates min 1 HypothesisCandidate)
```

In the last step of the approach depicted in Figure 5.3 the road users from a real-world dataset that are already associated to roads of an intersection are represented in the knowledge base. For that purpose, the knowledge base contains the *Participant* TBox as illustrated in Figure 5.6. Let us assume that all traffic participants of the dataset are associated to roads of an intersection (using the approach presented in Section 3.4). Furthermore, using the `Hypothesis` *TBox* we assume that road pairs are mapped to the different variants of SSCS that potentially exist on the intersection. Lastly, we assume that road users that are involved in a SSCS are extracted for every hypothesis variant. Then, for every concrete scenario, a road user is assumed to be the *ego participant* and the other

road users the *challengers*, represented as `EgoParticipant` and `ChallengerParticipant` in the knowledge base. The road users are associated to a `HypothesisCandidate` via the property `is_participant_for`, which is mapped to the specific SSCS variant via the property `is_candidate_for` of the `Hypothesis` *TBox*. The `Participant` *TBox* also contains subclasses of the `is_participant_for` property for the ego and challenger case. For this definition, we use SWRL rules[2] as shown in Listing 5.8. For instance, an individual of `EgoParticipant` that is associated to a `HypothesisCandidate` via the `is_participant_for` property can also be associated to that individual via the `is_ego_participant_for` property as shown in Listing 5.8. Since the `has_participants` is the *inverse* of the `is_participant_for` property, after reasoning we can find the ego participant and the challengers via the properties `has_ego_participants` and `has_challenger_participants` of a scenario candidate.

Listing 5.8: The SWRL rules for the definition of the `is_ego_participant_for` and `is_challenger_participant_for` properties.

```
1   EgoParticipant(?t) ^ is_participant_for(?t, ?v)
2      -> is_ego_participant_for(?t, ?v)
3
4   ChallengerParticipant(?t) ^ is_participant_for(?t, ?v)
5      -> is_challenger_participant_for(?t, ?v)
```

5.4 Extraction and transformation of OpenDRIVE intersection

In the last section, the different ABox entities of the knowledge base were introduced and the iterative approach for extending the knowledge base to ultimately find and validate hypotheses of SSCS in an intersection. In the following, it is shown how to extract the required topological information and the geometric representation of the intersection's digital representation given in OpenDRIVE format serving as the ABox sources. The overall approach is depicted in Figure 5.7, which is described in the following.

Since the OpenDRIVE specification utilizes the eXtensible Markup Language (XML) syntax, road networks in OpenDRIVE format are represented as such. To extract the connectivity of an intersection, a XSL transformation is applied.

In OpenDRIVE, every intersection is represented by a `junction` and every road by a `road` entity. A `road` entity has a `junction` attribute to associate it with a `junction` and contains linking information to other road entities. For instance, the road `300015` of the junction `300000` is shown in Listing 5.9[3]. Its predecessor is the road `100000` and successor the road `279000`.

[2] Instead of using SWRL rules, one could also employ *rolification* to define this relationship [112], but SWRL rules were chosen for the sake of clarity.

[3] For readability reasons, only excerpts from the XML files are shown.

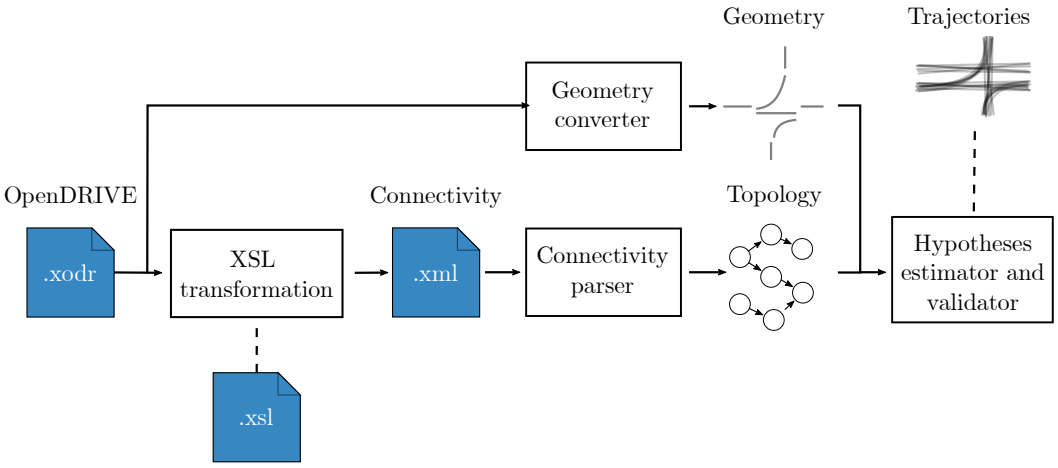


Figure 5.7: The intersection geometry and topological information is extracted from the OpenDRIVE map to estimate SSCS hypotheses. These hypotheses are validated with trajectories.

Listing 5.9: A road definition in OpenDRIVE with linking information to other road entities.

```
<road ... id="300015" junction="300000">
  <link>
    <predecessor elementType="road" elementId="100000" ... />
    <successor elementType="road" elementId="279000" ... />
  </link>
  ...
</road>
```

The road's link information about the predecessor and successor is used to extract the intersection's connectivity with the XSL transformation defined in 5.10. Instead of extracting all intersections at once, the transformation is applied to single `junction` entities. This allows performing the transformation in parallel. For that purpose, the `$junction` parameter in Listing 5.10 is utilized. For every `road` that is part of the intersection defined by `$junction`, a `node` element will exist in the resulting XML file. Every `node` element has an attribute `id` denoting the identifier of the road, `predecessor` the identifier of the preceding and `successor` for the following road.

Listing 5.10: The XSL transformation to extract an intersection's connectivity.

```
<xsl:for-each select="road[@junction=$junction]">
  <node>
    <xsl:attribute name="id">
      <xsl:value-of select="@id" />
    </xsl:attribute>
    <xsl:attribute name="successor">
      <xsl:value-of select="link/successor[@elementType='road']/@elementId" />
    </xsl:attribute>
    <xsl:attribute name="predecessor">
      <xsl:value-of select="link/predecessor[@elementType='road']/@elementId" />
    </xsl:attribute>
  </node>
```

```
13 </xsl:for-each>
```

The final XSL file for the transformation is depicted in Listing 5.11. Since the root node of the OpenDRIVE file is the `OpenDRIVE` element, we will match it and create a new `intersection` element with the `id` specified by `$junction`. The latter is a parameter of the XSL transformation defined in Listing 5.11:2. The `connectivity` element of the `intersection` contains the content from Listing 5.10.

Listing 5.11: The XSL transformation to extract an intersection's connectivity.

```
1  <xsl:stylesheet version="1.0" xmlns:xsl="http://www.w3.org/1999/XSL/Transform">
2    <xsl:param name="junction" />
3    <xsl:template match="/OpenDRIVE">
4      <intersection>
5        <xsl:attribute name="id">
6          <xsl:value-of select="$junction" />
7        </xsl:attribute>
8        <connectivity>
9          <!-- content from  Listing 5.10  -->
10       </connectivity>
11     </intersection>
12   </xsl:template>
13 </xsl:stylesheet>
```

Applying the transformation to the OpenDRIVE file for the DLR AIM research intersection generates a XML file partly depicted in Listing 5.12. Every road of the intersection is represented as a `node` element with attributes linking to the successor and predecessor nodes.

Listing 5.12: The extracted connectivity of the DLR AIM research intersection.

```
1 <?xml version="1.0"?>
2 <intersection id="300000">
3   <connectivity>
4     <node id="300010" successor="200000" predecessor="179000" />
5     <node id="300011" successor="180000" predecessor="100000" />
6     ...
7     <node id="300025" successor="279000" predecessor="179000" />
8   </connectivity>
9 </intersection>
```

To create hypotheses of SSCS a transformation is applied to the XML with the intersection's connectivity to represent the intersection's topological information as a DAG serving as the ABox. In fact, the definition of an intersection as a subset of a network $i \subset N$ and $N = \{W, \phi\}$ matches the definition of a DAG $G = \{V, E\}$ with V as the vertices and E as the edges. The vertices are the roads and the edges are represented in the connectivity set.

To transform the XML file with the intersection's connectivity to a DAG denoting the intersection's topological information, the `Connectivity parser` implements Algorithm 2. In a nutshell, the Algorithm 2 converts the list of nodes from the XML file to a graph of nodes and adds two virtual nodes denoting the start and end of the graph. The latter enables to easily identify incoming and outgoing roads of the intersection. This work employs the `NetworkX` [94] package since it allows managing and processing graphs, such as a DAG, in Python.

Algorithm 2 Extract the topological relationship as a DAG using the intersection's connectivity.

Input: Connectivity ϕ
Output: Topology $D(W, \phi)$
// Initialization:
$N_{\text{start}}, N_{\text{end}}$:= virtual start and end node
$D := \{\{N_{\text{start}}, N_{\text{end}}\}, \{\}\}$ *// initialize DAG*
// Graph setup
for all n in ϕ **do**
  // add nodes to DAG
  $D_W := D_W \cup \{n_{predecessor}, n_{id}, n_{successor}\}$
  // create links for predecessor node
  $D_\phi := D_\phi \cup \{(N_{\text{start}}, n_{predecessor}), (n_{predecessor}, n)\}$
  // create links for the successor node
  $D_\phi := D_\phi \cup \{(n, n_{successor}), (n_{successor}, N_{\text{end}})\}$
end for

Applying the Algorithm 2 to the example intersection's connectivity XML file gives the DAG depicted in Figure 5.8. The two blue nodes represent the virtual start N_{start} and end N_{end} nodes represented in the `Connectivity` TBox as `StartNode` and `EndNode`. The other nodes stand for roads of the intersection. The arcs between the nodes denote the intersection's connectivity represented via the `is_succesor` property in the TBox.

Representing the intersection as a DAG as shown in Figure 5.8 has several advantages. At first, we have a clear visual representation of how road users cross the intersection. That is, all paths from the start node to the end node represent a possible driving route through the intersection, while assuming normative behavior of traffic participants. Moreover, grouping roads as defined in (5.1) becomes now straightforward. By dividing the graph into layers from the start node to the end node, the first layer would be the incoming roads W_{in}, the second the connecting roads $W_{between}$ and the third layer the outgoing roads W_{out}. This allows verifying the results after reasoning on the ABox visually.

The geometry of the intersection's roads as depicted in Figure 5.7 are required to create hypotheses. For that purpose, the OpenDRIVE reference lines are employed, which are also used in Section 3.4 for route estimation. In OpenDRIVE, every geometric entity is represented by different geometric shapes, such as arcs, curves or lines or combinations of them. The `Geometric converter` uses the `opendrive2lanelet` [68] Python package to convert reference lines into, so-called, `LineString` objects. That is, every reference line is represented by a series of points by sampling the OpenDRIVE geometric entity definition. Since the widely used `GeoPandas` [113] natively supports `LineString` objects, the `Geometric converter`, which is also made publicly available [114], provides the road network as a `GeoDataFrame`. The road geometries of the example intersection are depicted in Figure 3.25 in Section 3.4.

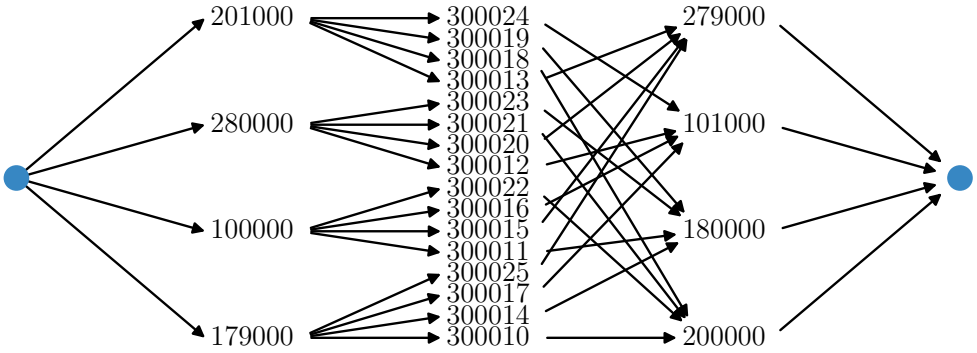


Figure 5.8: The extracted topology of the DLR AIM Research Intersection depicted as a Directed Acyclic Graph (DAG).

5.5 Results and discussion

The knowledge base of SSCS allows to formally describe and systematically identify SSCS scenarios in an urban intersection. This chapter will show the result of building the SSCS knowledge base using the digital representation of the DLR AIM research intersection in OpenDRIVE format and real-world data collected in that area. In particular, the a priori inferred hypotheses will be used to find *crossing scenarios* for demonstrating the feasibility of the presented approach. Moreover, it will shown be that the SMoS PET (see Section 2.4.3) can be used to validate hypotheses of crossing scenarios.

5.5.1 Dataset

For evaluation, a dataset is employed that is collected on the 12th April 2023 from 10:00 until 24:00 that contains only motorized traffic participants. The route estimation approach presented Section 3.4 was applied to associate the road users to the intersection's roads. The dataset was clustered, to get road pairs that overlap in time for each crossing hypothesis. Road pairs having a PET smaller than four seconds were filtered out. These pairs are the candidates of SSCS hypotheses used for validating SSCS hypotheses.

The traffic density distribution of the dataset is depicted in Figure 5.9 on an hourly basis with traffic participants associated to the different routes. For reasons of readability and clarity, the routes derived from the OpenDRIVE topology are mapped to the four cardinal directions `East`, `North`, `South`, `West` to describe the entry and exit directions of road users. For instance, the route $[201000, 300018, 200000]$ is represented as `WestEast` since road users enter the intersection from the west and leave it to the east. The road users that enter and leave the intersection from the same direction are grouped into *other* since the roads are not part of the crossing scenario hypotheses defined in (5.10).

The traffic density in the used dataset increases from 10:00 of approx. $2,800$ number of road users until reaching its peak at 13:00 of approx. $3,700$ road users. Afterwards, the density steadily decreases to the overall lowest density at 23:00 of 90 road users. Since

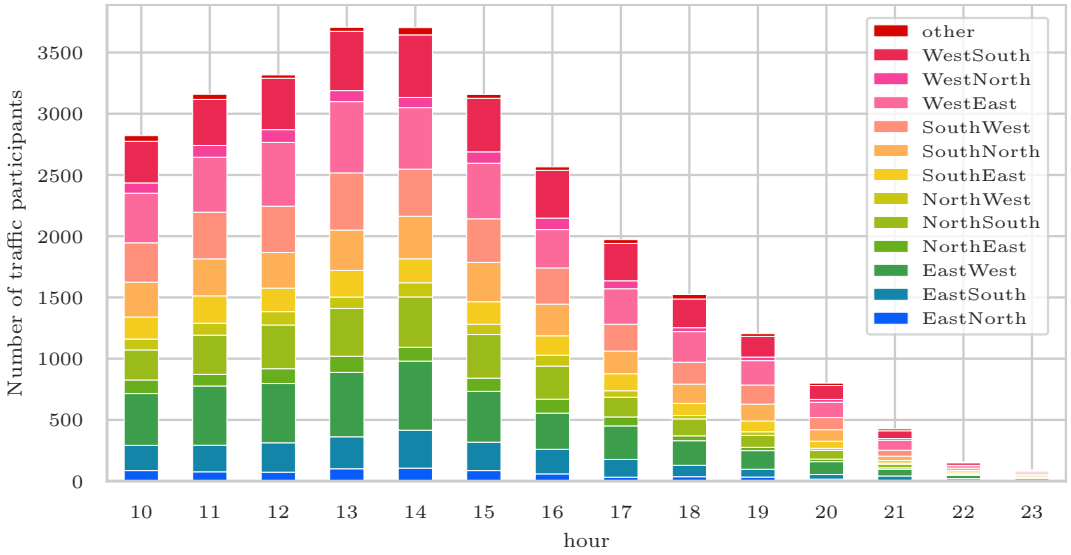


Figure 5.9: The evaluation dataset contains motorized traffic participants crossing the intersection on different routes. The routes are represented by the cardinal directions in which road users enter and leave the intersection. The four routes in which a road user enters and leaves the intersection in the same direction are aggregated in *other*.

the aim is to validate crossing scenario hypotheses, the dataset should contain road users associated to the roads that are part of scenario candidates defined in (5.10). Figure 5.9 clearly shows that the dataset contains a vast number of road users per route. Thus, it can be assumed that if a crossing hypothesis is valid, the dataset should contain pairs of traffic participants that are associated to the corresponding routes.

The methodology of map-driven maneuver identification proposed in Chapter 4 also enables to analyze the maneuvers conducted by traffic participants while crossing the intersection. In Figure 5.10 the distribution of maneuvers conducted by road users is illustrated. The majority of road users cross the intersection straight ($n = 13852$), followed by right ($n = 7191$) and left turns ($n = 7160$), with a minority of road users ($n = 400$) performing u-turns.

5.5.2 Results

The methodology proposed in this chapter allows identifying combinations of routes and thus space-sharing conflict scenarios hypotheses. Since the traffic flow is regulated via, e.g., traffic lights and traffic rules, not all hypotheses are valid. In the following, an example will be provided of how to identify hypotheses of SSCS and verify these hypotheses using the dataset previously described, specifically for the crossing scenario.

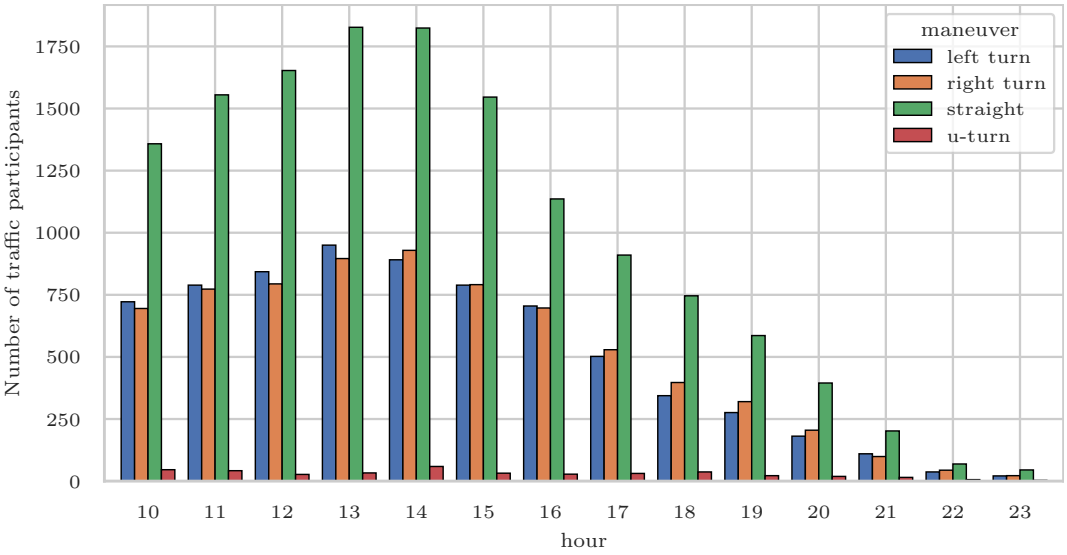


Figure 5.10: The dataset used for evaluation contains motorized traffic participants crossing the intersection by performing different maneuvers.

Classification of roads

To create hypotheses of SSCS it is mandatory that the roads of the intersection are classified correctly. Otherwise, the knowledge base may contain non-existent hypotheses. The result can be visually verified after reasoning on the ABox and TBox using the digital representation of the intersection. Figure 5.11 shows the results for the three road types using the DAG representation of the intersection. The *incoming*, *connecting* and *outgoing* roads are depicted from left to right, while the classified roads are highlighted in red. The figure indicates that all roads are correctly classified.

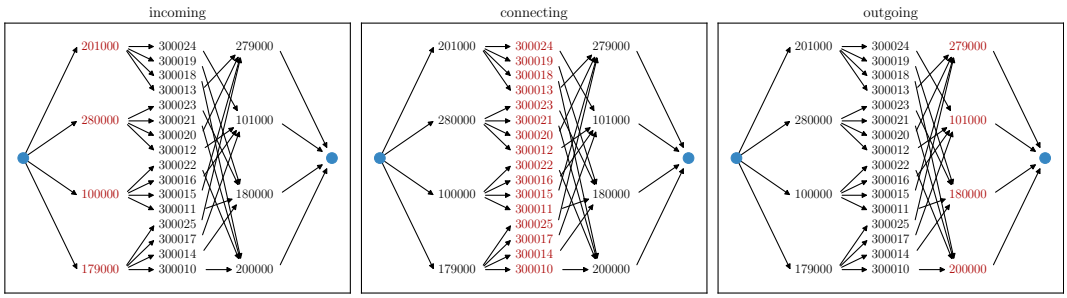


Figure 5.11: The roads of the intersection are classified using the `Connectivity` TBox.

Identification of crossing scenarios

To identify and extract concrete crossing scenarios from real-world data, hypotheses are defined in Section 5.3. To find hypotheses of crossing scenarios on the AIM Research in-

tersection, the OpenDRIVE reference line for each road is used. Figure 5.12 illustrates the results. The left panel of Figure 5.12 shows the intersection with the reference lines and the crossing points of all potential scenario candidates ($n = 38$). By applying (5.5) invalid scenarios are filtered out using the road topology and the information about a road's predecessors and successors. Thus, the final set of crossing hypotheses is

$$\begin{aligned} H_C = \big\{&(300015, 300014), (300016, 300014), (300017, 300015), \\ &(300018, 300014), (300018, 300015), (300018, 300017), \\ &(300019, 300016), (300019, 300017), (300020, 300016), \\ &(300020, 300017), (300020, 300018), (300020, 300019), \\ &(300021, 300014), (300021, 300015), (300021, 300016), \\ &(300021, 300019)\big\}. \end{aligned} \tag{5.10}$$

and contains $n = 16$ pairs of routes (see Figure 5.12 (right)). Note that all points are removed where different roads merge together or dissolve and only points in the center of the intersections are left.

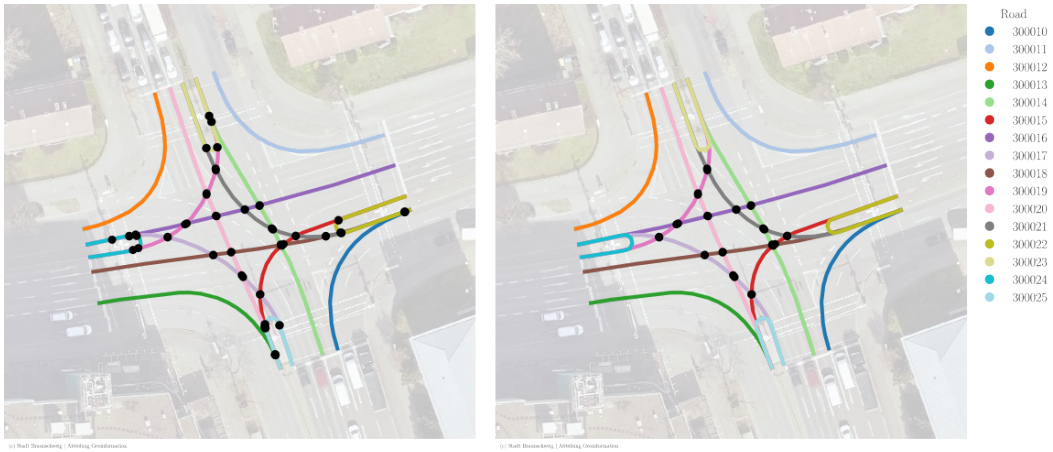


Figure 5.12: The roads crossing each other are candidates for crossing SSCS. *Left:* All crossing points on the intersection. *Right:* The relevant crossing points after filtering using the road topology.

Validation of crossing scenarios

The set of road pairs contains all potential hypotheses of crossing scenarios. As illustrated in Figure 5.7 the hypotheses need to be validated. For this purpose, concrete scenarios were extracted from real-world data. This topic will be discussed in Chapter 6. Figure 5.13 shows the hourly distribution with a total number of 562 scenario. The left panel of the diagram shows the distribution of concrete scenarios throughout the day, while the right side shows the total number of incidents per scenario hypothesis in logarithmic scale. Note that the right diagram serves as the legend of the left one. Both diagrams clearly

show that the most crossing scenarios occur for the *WestNorth-EastWest* hypothesis with 428 incidents and thus 76% of all scenarios. The other scenarios primarily occur between 10:00 and 17:00.

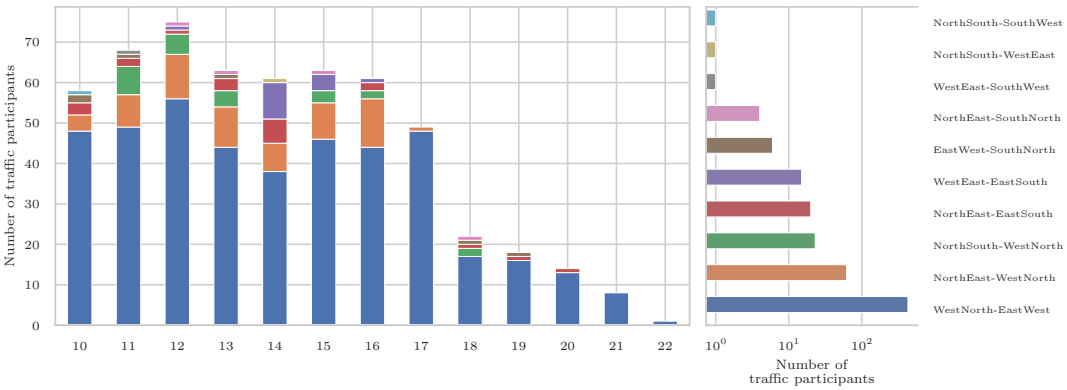


Figure 5.13: The distribution of crossing scenarios in the dataset. *Left:* The number of scenarios throughout the day (scenarios are color-encoded). *Right:* The number of traffic participants per crossing scenario in logarithmic scale.

Another perspective on the experimental results is given in Table 5.1, showing the number of concrete scenarios for different routing pairs (crossing hypotheses). The crossing scenario hypotheses defined in (5.5) are highlighted and the pairs of routes without any associated road users are denoted as "-". Regarding the validation of scenario hypotheses, the table helps to identify valid and invalid hypothesis. That is, cells in the table that are not highlighted and contain "-" do not represent crossing scenario hypotheses or have any associated road users. The highlighted cells that contain "-" are hypotheses of crossing scenarios but without any associated road users and could be assumed invalid hypotheses. The highlighted cells without a "-" represent the number of concrete scenarios for this hypothesis and thus could be assumed valid.

The experimental results shown in Table 5.1 indicate the following. There are 10 hypotheses with at least one pair of road users involved in the crossing scenario. Hence, there are 6 hypotheses without any associated road users. That is, for this experimental setting, and for the use case of identifying crossing scenarios, the original number of road pairs with crossing reference lines could be reduced from 38 to 16 (the crossing scenario hypotheses) by considering the road network topological information, down to 10 scenarios (the valid hypotheses) using real-world trajectory data, reducing the overall number of relevant road pairs by 73.68%.

The right diagram of Figure 5.13 and the Table 5.1 also shows that there are three crossing scenario hypotheses having a single incident. If they are really valid crossing scenarios, why is there only a single incident of each scenario? In the following, we will analyze these scenarios using the trajectory data and video material to verify if theses hypotheses are valid.

The trajectories of the road users involved in the three scenario hypotheses with single incidents are depicted in Figure 5.14. The figure illustrates the spatial relationship of the

Table 5.1: The crossing scenarios identified in the dataset with at least two traffic participants involved having a PET smaller than four seconds. The hypotheses of crossing scenarios derived from the road network are highlighted.

	EastSouth	EastWest	SouthNorth	SouthWest	WestEast	WestNorth
EastSouth	-	-	-	-	-	-
EastWest	-	-	6	-	-	-
NorthEast	23	-	4	-	-	63
NorthSouth	-	-	-	1	1	23
SouthWest	-	-	-	-	-	-
WestEast	15	-	-	1	-	-
WestNorth	-	459	-	-	-	-

road users. It is clearly visible that both trajectories intersect. To determine the cause of the incident, the video-material provided by the cameras was analyzed.

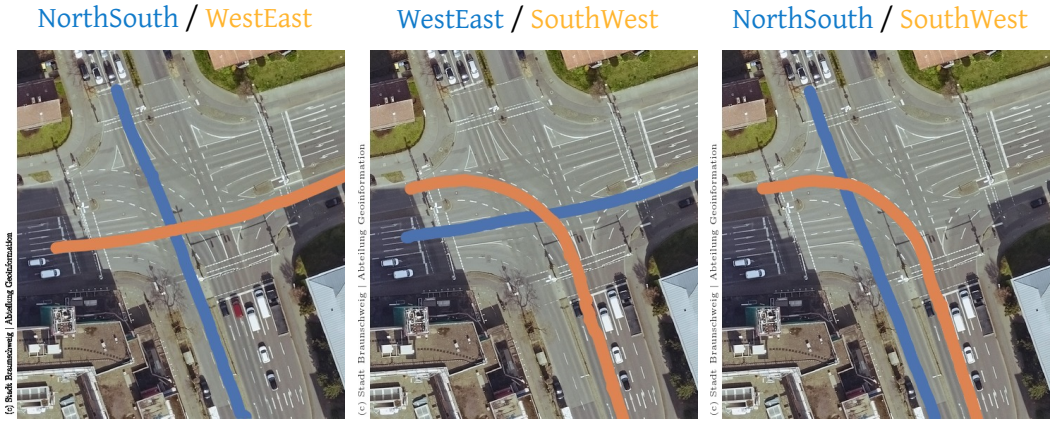


Figure 5.14: The three single incidents of crossing scenarios. The trajectories of both traffic participants involved in this scenario are depicted.

The first concrete scenario depicted in Figure 5.14 occurred because of a firefighter-vehicle crossing the intersection (see Figure 5.15). The firefighters enter the intersection from the west (see Figure 5.15 left) and exit it towards the east. The other road users wait and continue their route, crossing the path of the firefighter-vehicle two seconds later (see Figure 5.15 right).

The second concrete scenario depicted in Figure 5.14 occurred because of a police vehicle crossing the intersection (see Figure 5.16). The vehicle enters the intersection from the west (see Figure 5.16 left) and exit it towards the east. The other road users wait and continue their route, crossing the path of the police vehicle three seconds later (see Figure 5.16 right).

The third concrete scenario illustrated in Figure 5.14 shows a vehicle (white) crossing the intersection violating the red-light rule. Three situations (t_1, t_2, t_3) of this scenario are

Figure 5.15: Three situations from the crossing scenario, in which a firefighter vehicle crosses the intersection from west to east while other road users wait before continuing their paths.

Figure 5.16: Three situations from the crossing scenario, in which a police vehicle crosses the intersection from west to east while other road users wait before continuing their paths.

depicted in Figure 5.17 for three cameras pointing in different directions of the intersection (*North*, *South*, *East*). At t_1 the white vehicle is visible in the image of the *South* camera waiting to cross the intersection. Note that vehicles crossing the intersection straight from south as well as both left-turn lanes have a red light. At t_2 the vehicles crossing the intersection straight from south have a green signal and thus already entered the intersection. The white vehicle also enters the intersection to perform a left-turn. Note that the oncoming vehicles also have a green light and are allowed to cross the intersection. For instance, the vehicles waiting at t_1 in *North* are leaving the intersection at t_2 in *South*. At t_3 the white vehicle stands on the intersection, but on one of the two oncoming lanes, giving way the oncoming vehicle. In fact, the white vehicle needs to break sharply to avert a collision with the oncoming vehicle. This is also visible in Figure 5.18 showing the velocity profile of both road users[4]. The white vehicle (SouthWest) starts accelerating at *11:58:25* (one second before t_2 in Figure 5.17) until *11:58:27* up to a velocity of approx. $20\frac{km}{h}$. At *11:58:31* the white vehicle starts to decelerate sharply to a full stop in 1.5 seconds, preventing a collision with the oncoming vehicle (NorthSouth). It is worth noting that even though the incoming vehicle had a clear view of the white vehicle, it did not reduce its speed significantly or perform any evasive maneuver.

[4]The trajectory of the road user NorthSouth is available for a limited duration of the scenario.

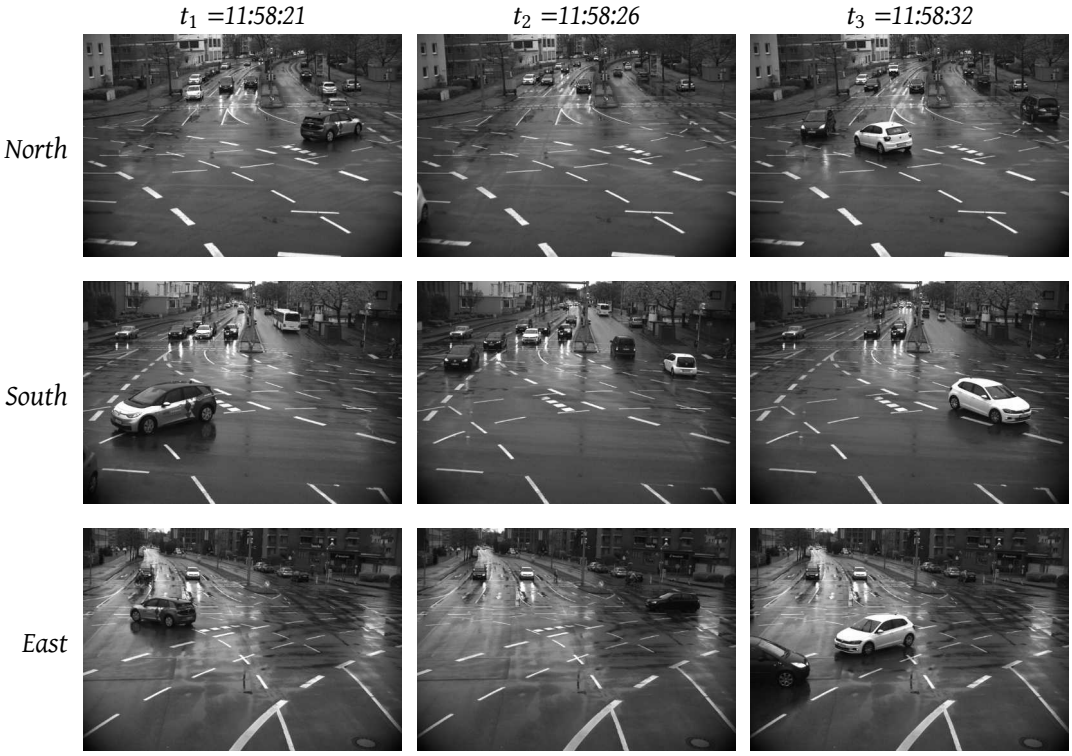


Figure 5.17: The images from three cameras pointing in different directions (*rows*) for three situations (*columns*) of the crossing scenario NorthSouth / SouthWest with single incident. The white vehicle entering the intersection from the south violates the red light rule.

5.5.3 Discussion

The three presented concrete crossing scenarios are valid crossing scenarios in terms of our previous definition. So, one could conclude that the corresponding hypotheses are also valid and provide crossing scenarios, reading the Table 5.1 as it is: For every cell with at least one pair of road users, the hypothesis is assumed valid. However, it is arguable if this is generally acceptable because the three situations are atypical because they occur due to non-normative behavior of traffic participants. If the dataset of crossing scenarios should be used to train a system to automatically cross the intersection, unwanted behavior might be integrated into the model, *e.g.*, violating red light rule. However, this situation might help to develop AVs that are able to safely and comfortably cross intersections in future mixed traffic environments, using this scenario as an example of traffic rule violation by human road users. An AV even have to cope with such types of atypical behavior, especially if the available traffic infrastructure is shared among humans and autonomous systems. This topic is still investigated extensively in academia, particularly scenarios involving vulnerable road users [115][116]. For that use case, the three single incident hypotheses could not be classified as valid since it would not be able to identify

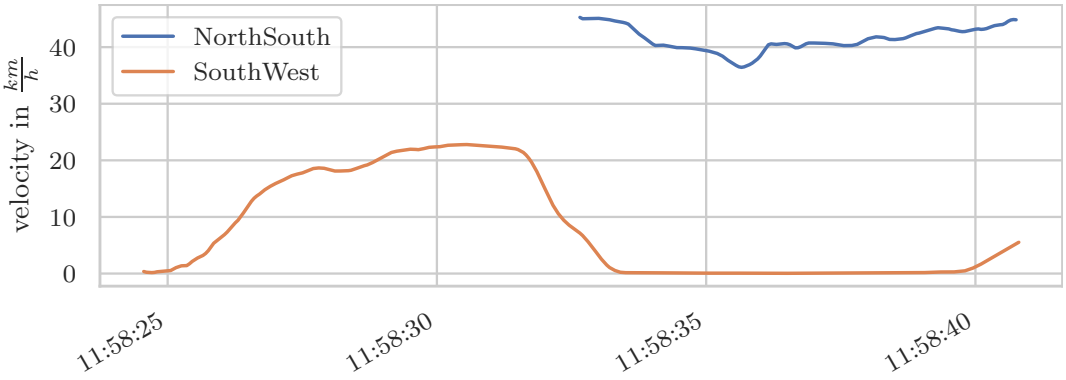


Figure 5.18: The velocity profile of both road users engaged in the concrete crossing scenario NorthSouth / SouthWest as the result of a red-light violation.

them using the presented approach. Instead, the hypotheses could be classified as *normative* or *non-normative*. That is, the three hypotheses with single incident are *non-normative*, while the other valid hypotheses are *normative*.

The same classification would apply if the scenario knowledge base is used as foundation for traffic flow analysis to, for instance, adapt green light phases [117]. In that case, the hypotheses would be assumed invalid to ensure a solid and valid data basis for analysis. The concrete scenarios of the other crossing scenarios are used to analyze and potentially adapt the green light phases to, *e. g.*, reduce the number of critical encounters, or increase the overall throughput.

The experimental results presented so far lead from the assumption that a crossing hypothesis is valid if there are scenario candidates having a PET lower than a certain threshold. That is, all scenario candidates having a PET higher than this value are not considered, thus affecting the overall number of scenarios. This work chooses a PET of four seconds for demonstration purpose. This value guarantees that the set of crossing scenario candidates does not only include close encounters. In fact, if a PET value of, for instance, two seconds as in [118], is chosen the three previously presented incidents would not be identified. Hence, choosing the PET threshold w.r.t the use case is recommended.

5.6 Summary

In the near future, automated vehicles will become available and share the available traffic infrastructure with human-driven cars. Thus, they not only have to pursue their driving task autonomously, which is already a complex task, but also need to cope with unforeseen situations, e.g., as a result of a red light violation. Since the automated vehicle is in conflict with other road users, the system must be able to perform situation-dependent strategies. For the development of such strategies, the understanding of human behavior is vital — especially in rare situations.

Scenarios occuring in the real-world help to analyze the human behavior and thus enable to develop mentioned strategies. In this chapter, a methodology is proposed to sys-

tematically find scenarios in which road users compete for the use of infracture, the space-sharing conflict scenario (SSCS) presented in Section 2.4. The overall approach aims at building a knowledge base of SSCS by formally defining hypotheses of SSCS that potentially exist on an urban intersection. For that purpose, the knowledge base contains different TBoxs, each focusing on specific entities in this context. In fact, one TBox represent the topological information of an intersection in the knowledge base. This is used in the second TBox to define hypotheses of SSCS. Since this step requires adding individuals to the knowledge base, SPARQL queries were defined to find pairs of roads matching with the definitions of SSCS. The third TBox allow representing road users that are associated to roads of the intersection in the knowledge base. Since the overall approach is based on a digital representation of an intersection in OpenDRIVE format, it was shown how to extract the required information from that digital map. This information serves as the ABoxs of the knowledge base.

The presented methodology was validated using the AIM Research Intersection and a dataset of motorized road users to identify crossing scenarios. That is, road users, the scenario candidates, are identified that are involved in the extracted SSCS. Those candidates and the SMoS PET are used to validate scenario hypotheses. The results has shown, that choosing the PET value is critical for hypothesis validation. In particular, choosing a use case related PET value is proposed to address the different requirements and goals.

Chapter 6

A modular platform for scenario mining

The Scenario mining platform (Scenimini) combines the results of Chapter 3, Chapter 4 and Chapter 5 into one platform for the purpose of scenario identification, extraction and analysis, addressing the O.II. In the following, an overview is given of how Scenimini is organized. Moreover, the scenario mining process is defined. It serves as the basis for the realization of the Scenario mining pipeline (SMP) as a key component of Scenimini for scenario extraction. The following chapter will also present the Python library Traffic situation analysis and interpretation (TASI) in Section 6.3.2 used by the SMP for traffic data management and analysis and the ScenORM in Section 6.3.1 as a scenario-driven ORM for an object-oriented and scenario-oriented view of traffic data management in a RDB. The chapter Section 6.4 concludes with an overview of recent and current use-cases and examples on the utilization of Scenimini for traffic analysis in the urban domain, rural area and on the highway.

6.1 Scenario mining platform architecture

Scenimini is a collection of components realized as a distributed system. This work follows the established architectural style to logically divide applications into layers [119] and organize Scenimini in three tiers as illustrated in Figure 6.1. Note that the arrows visualize the data flow between the components.

Data tier As for all data processing processes, data is the main building block. This also holds for Scenimini with the *data tier* representing various data sources. The *GeoServer*[1] is an open-source server for sharing geospatial data in standard protocols such as the Web Feature Service (WFS) or the Web Map Service (WMS). It is employed by the SMP services to solve tasks that require a digital representation of the road network. Furthermore, the applications use the GeoServer in combination with trajectory data for visualization purpose. The second component is *traffic data*, which is obtained for a period of time during test campaigns via floating vehicles or quasi-stationary infrastructure or continuously by stationary infrastructure (see Section 1.1.3). The acquired traffic data includes trajectory data of traffic participants obtained from different sources such as the VuT or traffic infrastructure, as well as weather information and video data. The third component *Scenario storage* stores the scenario-related results extracted from traffic data by the components

[1] https://geoserver.org/

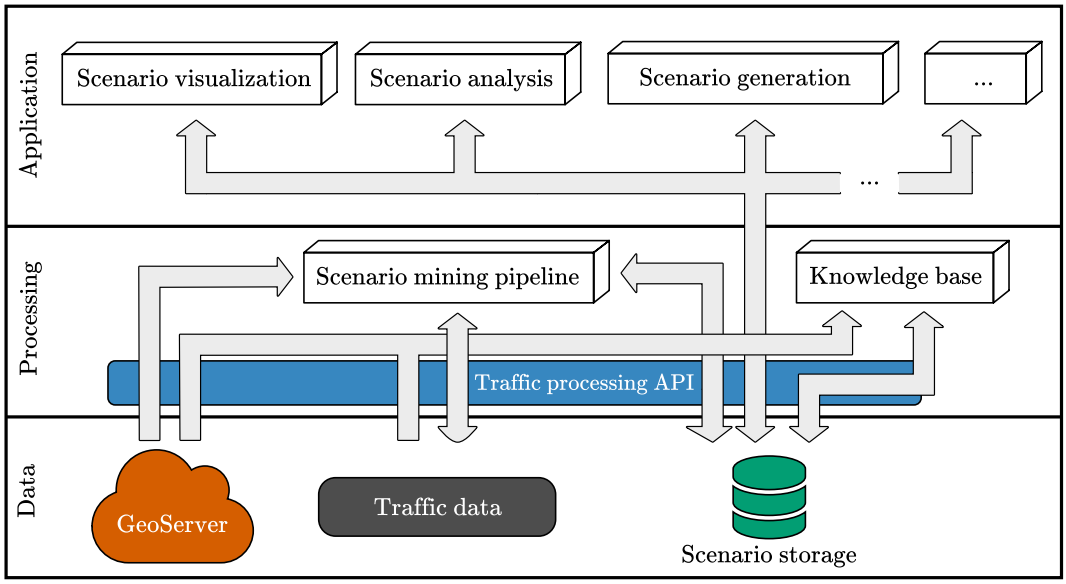


Figure 6.1: The three tier architecture of the scenario mining platform.

of the *processing tier*. In fact, its the representation of traffic data using a scenario-oriented view through different variants of scenarios, maneuvers and primitives, as shown in Chapter 3 and Chapter 4.

Processing tier The middle layer of Scenimini is the *processing tier* and its main purposes are the processing of traffic data for the sake of a scenario-oriented view of traffic data and providing access to the data layer via standardized interfaces. The SMP which will be introduced in Section 6.2 is a specialized data processing pipeline dedicated to the extraction of scenarios from traffic data. It results are employed by the Knowledge base that was introduced in Chapter 5. The SMP and the Knowledge base access the components of the data tier through the Traffic Processing API (TAP-API). It provides standardized access to the data tier via specialized interfaces and a toolkit for traffic data management and analysis which will be presented in Section 6.3.

Application tier The topmost layer of Scenimini is composed of applications. Its components use the results of the *processing tier* according to the specific use case. For that purpose, the applications use the interfaces provided by the TAP-API. The applications vary and are dependent on the use case. This includes the visualization of extracted scenarios, perform qualitative and quantitative scenario analyses, or for the purpose of scenario (re)simulation in simulation environment.

The descriptions thus far give a general outline of the tiers and components of Scenimini. The overall purpose of Scenimini and the relationship between the different tiers and components, is, however, best described through an example. Let us reconsider the use case from Chapter 5, which was about the analysis of the behavior of traffic partici-

pants at an urban intersection during crossing scenarios. The *data tier* provides the trajectories of traffic participants that are required to find the actual crossing incidents. Moreover, the digital representation of the road network, provided by a `GeoServer`, is used to estimate each traffic participant's route while traveling through the intersection. The actual fusion of traffic data and the digital map of the road network, as well as the data enrichment for the sake of scenario extraction takes place within the SMP. The crossing scenarios extracted by the SMP are persistently stored in the `Scenario storage` of the *data tier* and used by the `Knowledge base` to validate the scenario hypotheses. The quantitative analysis of crossing scenarios conducted in Section 5.5.2 can be located in the *application tier*, while it uses the `Traffic processing API` to query the scenarios from the *scenario storage*. The same holds for the visualization of the traffic participants that are involved in the scenario on a satellite image of the intersection in Figure 5.14, and in video data acquired by the camera-based traffic infrastructure. The `Traffic processing API` provides access to the data as well as to the `GeoServer` hosting the satellite images.

6.2 Scenario mining pipeline

The Scenario mining pipeline (SMP) plays a vital role in Scenimini. It is located in the *processing tier* and connects various sources of information for the purpose of finding scenarios in traffic data. In the following, a several aspects of the SMP are highlighted. At first, an overview of the overall scenario mining process is given, *i.e.* how to structure traffic data in terms of scenarios. Then, it is shown how the SMP is organized for the purpose of scenario mining. Finally, this section will show how the SMP is configured and deployed.

6.2.1 Scenario mining process

The main purpose of the SMP is to organize traffic data in terms of scenarios. As scenario descriptions can vary, the scenario mining process usually depends on the scenario of interest. But, there are common processing tasks that are shared across the scenarios and are part of nearly all data processing tasks. In [87], a four-stage process was proposed for maneuver mining. In the following, this perspective is adapted for the process of scenario mining and divided into three primary tasks as illustrated in Figure 6.2, which will be briefly described in the following.

The first task involves the preprocessing of traffic data and represents the first two stages of [87]. Especially if the traffic data is collected in the real world, it is prone to errors and data preprocessing becomes mandatory. [99] This task involves filtering to reduce measurement outliers and smoothing to reduce measurement noise or adding missing information due to data inconsistencies. Furthermore, if traffic data should be fused with other data sources, it typically needs to be aggregated so that different data source share a common time base.

The second task is dedicated to add information to the available traffic data by, *e.g.*, data fusion. For instance, a scenario description usually specifies on which road traffic participants drive. For that purpose, the traffic data needs to be fused with information

from a digital representation of the road network via *map matching* served by a *GeoServer*. Hence, traffic data is enriched by linking road users to the digital representation of the road network. Another focus of this task is the semantical representation of traffic data in terms of primitives and the description of driving behavior on terms of maneuvers as shown in Chapter 3.

The third task structures traffic data in terms of scenarios. The aim is to identify the scenarios of interest based on the enriched traffic data and the semantical representation of the traffic participant's driving behavior based on primitives and maneuvers. This task also involves the analysis of scenarios and filtering according to specific parameters defined in the scenario description. For our previous use case with crossing scenarios, a relevant parameter is the PET. That is, scenarios may be selected where traffic participants interact within a specific PET range.

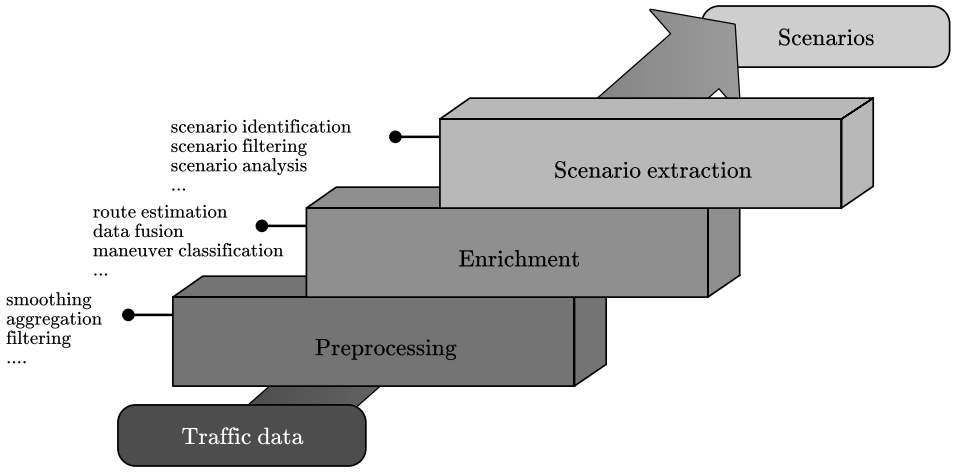


Figure 6.2: The overall process of scenario mining from real-world traffic data that is implemented in the scenario mining pipeline.

6.2.2 System architecture

The scenario mining process describes how traffic data is processed to find the scenarios of interest. This procedural view is transferred to an architectural view and the SMP is organized as a collection of tasks that is logically represented with the three layers (see Figure 6.3). Moreover, the tasks of the scenario mining process are encapsulated so that each of them work independent of the other tasks. This is due to the requirements that the SMP's functionality should be extendable during runtime and that the SMP should be used for various use cases and thus needs to be customizable. To achieve this, a service-oriented architecture (SOA) is employed for the SMP, which is thus, according to [119], a collection of loosely organized *tasks*, each encapsulated as a *service*. In the context of SOA, the term *loose coupling* typically arises and generally means that services provide an interface for their functionality to cooperate with others, instead of directly communicating with other services [120]. The major benefits of loosely coupling services are, according to [120], that

additional services can be easily integrated into the system by using the interfaces of other services, and that services can be used in other use cases where the service's functionality is required.

Each service of the SMP thus performs a specific *task* along the scenario mining process and, consequently, every layer of the SMP is composed of several *services* (illustrated as circles in Figure 6.3). That is, a service may smooth trajectories of traffic participants or associate traffic participants to roads of a road network. With each service having a dedicated purpose, the visualization of the scenario mining process from Figure 6.2 can be transformed to the arrow shown at the bottom of Figure 6.3. That is, the scenario mining process is realized by connecting the tasks of the SMP's services.

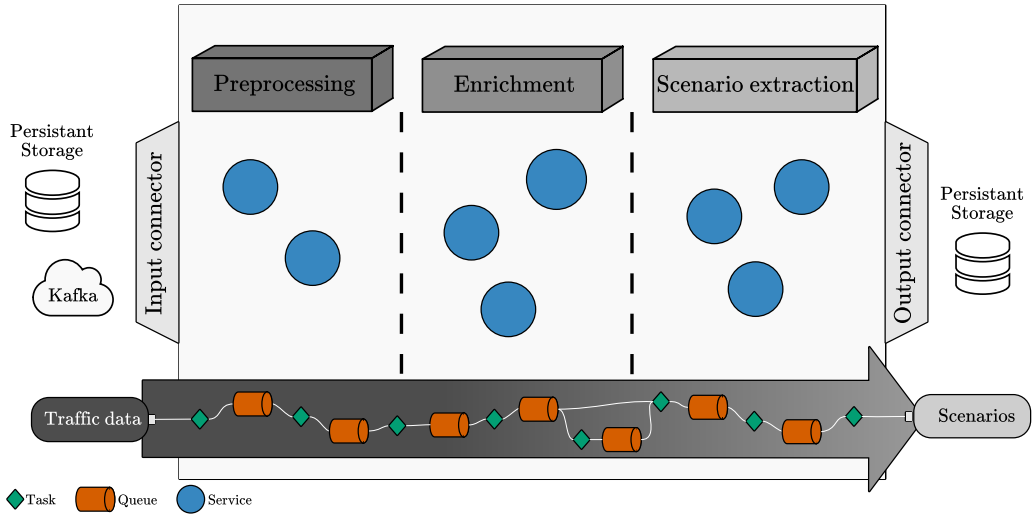


Figure 6.3: The realization of the scenario mining pipeline as a distributed system following a service-oriented architecture (SOA).

Input and output connectors Thus far, the scenario mining process was illustrated as a series of encapsulated tasks. However, if the process is considered as a data flow from traffic data to scenarios through services, it must be specified how data is flowing into and out of the SMP. To accomplish this, the SMP contains two sets of components: the *input* and *output connectors*.

The *input connectors* on the left side of Figure 6.3 are components that manage the information flow from persistent storages (*e. g.* an RDB) or from traffic infrastructure via Kafka into the SMP dependent on the use case. Let us consider the use case in which scenarios should be mined from historical traffic data that is managed in a RDB for a specific time range. In that case, the connector implements the three-phase extract-transform-load (ETL) process. In the first phase *extract*, the connector fetches the information for that time range from the RDB. In the second phase *transform*, the connector converts and properly prepares the information so that other services are able to process it. Finally, in the *load* phase, the service forwards the information into the SMP. Another use case is quasi-

live scenario mining, in which traffic data from an RDB is to be processed up to a point in time before the last available time (*lag time*). As in the previous use case, the connector needs to fetch traffic data from the database. But, it also needs to track the progress to verify that traffic data is forwarded up to the lag time and send new incoming data. The same use case could be based on information that is collected from traffic infrastructure and distributed via, *e. g.*, Kafka. In that case, the connector needs to aggregate information from Kafka, convert the data and forward it into the SMP.

The *output connectors* on the right side of Figure 6.3 are components that store the results in a persistent storage. Since the components of the *application layer* in Figure 6.2 as well as the *Knowledge base* use these results, several output connectors exist. The purpose of one connector is to store the details of the extracted scenarios in a RDB. Another connector saves videos of the scenarios for in-depth scenario analysis. Note that although the output connectors are on the right side of the SMP, they can be attached to services along the scenario mining process to, *e. g.*, allow storing intermediate results in a RDB.

Service communication The SMP is organized as a collection of loosely coupled services to realize the scenario mining process. Although services should work independent of other services, they can cooperate with others along the aforementioned process in the SMP. There are several ways discussed in the literature to establish communication between services in a distributed system with a SOA. Since the SMP should be highly flexible in terms of the services that realize the scenario mining process and fault tolerant in case of service failure, message-oriented communication between services is used. For that purpose, a MOM is employed that manages the communication between services via persistent message queues. This is also illustrated in the bottom arrow of Figure 6.3 where the tasks of the scenario mining process, that are encapsulated in services, are connected via queues. Every service that *publishes* information into the SMP (producer), sends messages to the MOM, which forwards the message to the destination queue. A service of the SMP that demands information from another service (consumer), *subscribes* to the named queue and will be notified if a new message arrives.

In this work, RabbitMQ is used as a lightweight and open-source MOM [121]. In RabbitMQ exists another entity along the message exchange process between consumers and producers, the *exchange*. That is, instead of directly forwarding messages from a producer to queues of consumers, they will be send to exchanges and RabbitMQ internally forwards them to bound queues using different strategies. An example is illustrated in Figure 6.4 in which the service `S.A` publishes messages and the three services `S.B`, `S.C` and `S.D` subscribes to those messages via the exchange `E.A`. Every service has its own queue (`Q.A`, `Q.B`, `Q.C`) bound to the exchange `E.A`. In this example, the messages published by `S.A` are forwarded via the exchange `E.A` to all services since it is a *fanout* exchange.

To meet the needs for a highly customizable SMP design, the *fanout* exchange variant is employed by default along the scenario mining process. This strategy allows to realize branches of the scenario mining process, with every branch having a specific purpose. As indicated in Figure 6.4, the services `S.B`, `S.C` and `S.D` process the message from `S.A` and could also publish their results, which in turn could be used by other services, thus forming

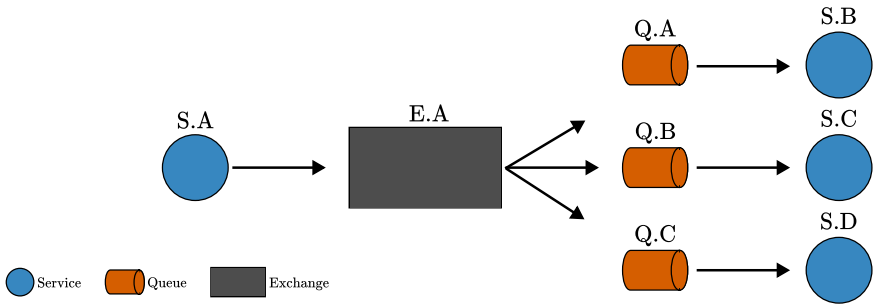


Figure 6.4: An example illustrating the usage of RabbitMQ fanout exchanges within the SMP along the scenario mining process.

a tree-like data flow. A branch of such tree could, for instance, be responsible for mining a specific scenario or for storing intermediate results.

The scenario mining process is composed of several tasks. Since the complexity of tasks can vary due to various reasons and thus the tasks can have different run times, individual tasks may become a performance bottleneck, limiting the overall throughput of the process. This is a critical issue due to the amount of traffic data that is continuously collected, particularly by traffic infrastructure and the need to process the data in foreseeable future. A typical approach to overcome bottlenecks is the concurrent processing of those complex tasks. In context of the SMP, multiple instances of a service are created to scale the SMP.

Let us consider the example illustrated in Figure 6.5 where the service `S.B` from Figure 6.4 is computationally intensive and poses a bottleneck in the overall process. To overcome this, the service is deployed three times as `S.B1`, `S.B2`, `S.B3`. Since a message should be forwarded to one of the three services, they consume messages from the same queue `Q.A`. That is, the messages will be scheduled such that a message is processed once by any of the three services. This strategy makes it possible to create multiple instances of a service in order to counteract performance bottlenecks.

6.2.3 Development and deployment

The SMP follows a service-oriented architecture (SOA), with every service implementing a specific task along the scenario mining process. Since the SMP is a collection of loosely coupled services, it must be specified which services are required to mine a specific scenario, how services exchange information and how they must be implemented so that they can be integrated into the SMP.

The scenario mining process illustrated in Figure 6.2 shows the process from traffic data to scenarios in linear format. The development of the SMP, however, follows an iterative approach of six phases. Figure 6.6 shows an abstract view on the process.

The first step of the overall process is to define the scenario of interest with all participating actors including their behavior and other environmental parameters. This definition is used to create the actual scenario mining process for that particular scenario.

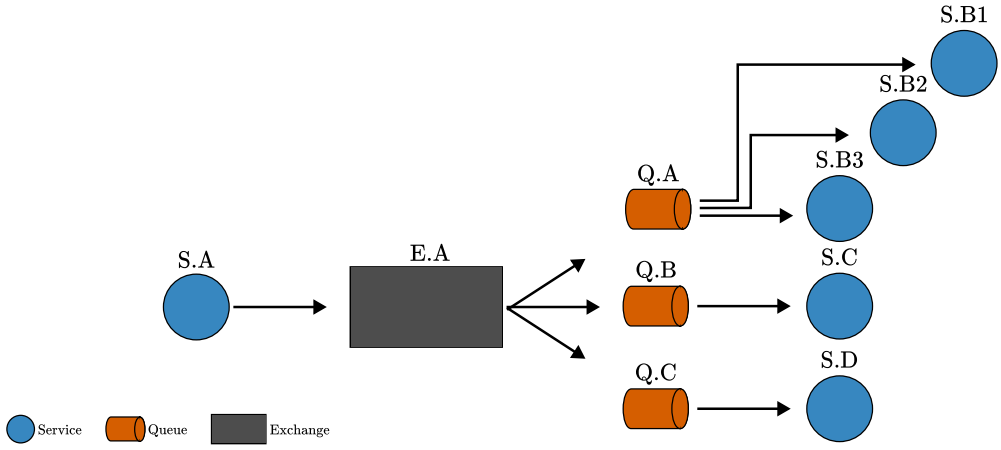


Figure 6.5: An example illustrating scaling the SMP by running multiple instances of a service and using RabbitMQ fanout exchanges.

That is, the aim is to break down the overall process into individual independent tasks. This also includes specify each task in detail as represented by the *Task definition* phase. For the definition of the tasks, the six-layer model of [67] can be employed. At first, the scenario-relevant parameters, attributes and relations for each layer are identified using the environmental model. Then, it must be specified how to obtain the information. This specification can be decomposed into dedicated steps, with each step representing a task of the scenario mining process. Since the SMP is decomposed of services, every task needs to be implemented as a service. To finally realize the scenario mining process based on the broken-down process and the list of services, the SMP is configured and deployed. The final step is the analysis of the extracted scenarios according to the use case. If the findings from this step result in further relevant scenarios or there are multiple scenarios that should be extracted, the process starts again from the beginning.

Since the services should be reusable for different use cases, they are registered in a central *service repository* upon definition (see Figure 6.8). That is, if a task is not represented by a service and thus not registered at the service repository, it needs to be implemented and registered by the service developer. This is indicated in Figure 6.8 as the connection between the *service implementation* phase and the *service repository*. In the *SMP configuration* phase, the service definitions are utilized to configure the SMP instance. On deployment, the service definitions are fetched from the *service repository* to create the SMP instance. In the following, some of the phases are described in more detail to give further insights into the process.

Service definition The first two phases of the overall process are concerned with the definition of the scenario that should be extracted from traffic data and breaking down the scenario mining process into individual tasks. This will be demonstrated in Section 6.4,

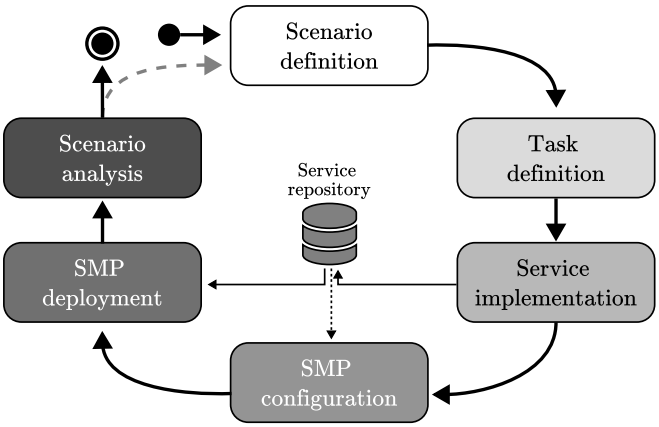


Figure 6.6: The six-phased development and deployment process for scenario mining using the Scenario mining pipeline (SMP).

where Scenimini and in particular the SMP is used for different use cases and environments.

For every task of the scenario mining process, a service needs to be defined. To rapidly create service prototypes and thus to realize or extend the SMP, the primary programming language that is used is Python. Moreover, Python provides a rich environment of libraries for data mining tasks. Since both RabbitMQ and Apache Kafka are supported as MOM, client libraries for communication are needed. In this work, *pika*[2] for the RabbitMQ client and *confluent-kafka*[3] are employed for that purpose. But, to hide the communication aspect from the service developer, reduce boilerplate code and thus to focus on the task implementation, the libraries internals are encapsulated. Figure 6.7 shows the classes that are used to realize a service using either Apache Kafka or RabbitMQ, while only relevant methods are illustrated.

The `ServiceMixin` defines the interface for a basic service, while the `KafkaService` and `RabbitMQServer` implement the interface for the actual MOM. A service developer can wrap the scenario mining task as a service by implementing the `process()` method. This method is called on message arrival of the MOM. Other details of the service such as the topic/exchange or queue name are defined via command-line arguments. This design was chosen to ensure that services can be used in different use cases.

A `KafkaService` or `RabbitMQService` only subscribes to the messages of the topic(s) or exchange(s) and does not publish any messages. This allows to realize services that do not add any additional information, such as services to persistently store the results. If a service should distribute results to other services, the `*Producer` services are utilized. Both services implement the `send()` method, which produces the messages given as argument w.r.t the MOM. As for the `*Service` classes, other communication aspects are hidden. For the RabbitMQ services, the `RabbitMQMixin class` defines the process to setup and man-

[2] `https://github.com/pika/pika`
[3] `https://github.com/confluentinc/confluent-kafka-python`

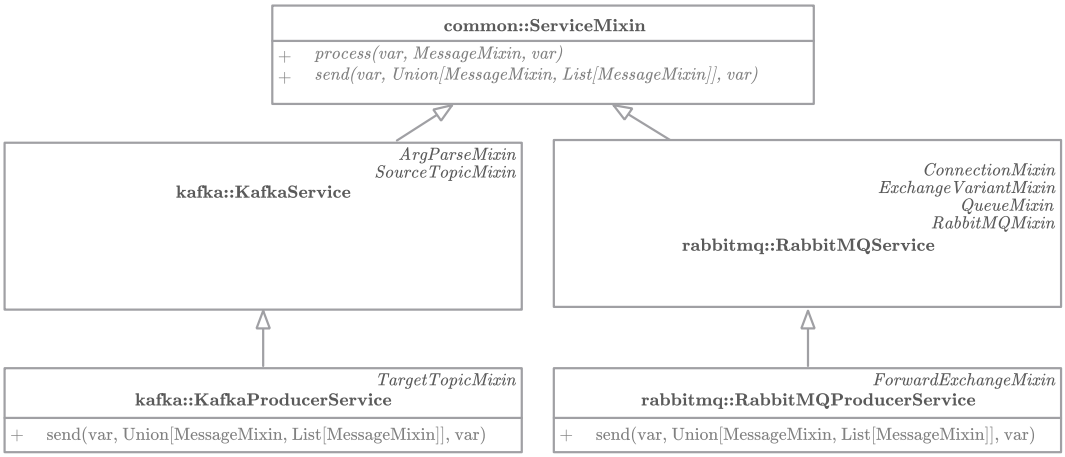


Figure 6.7: An excerpt of the class definitions to realize a service as subscriber and publisher using RabbitMQ as the MOM for message exchange.

age a communication channel with RabbitMQ based on *pika*. Moreover, the default configuration for exchanges and queues are defined in the `RabbitMQMixin` class and can be overwritten by the service developer and on deployment.

After implementing the task with any of the four `Service` classes, the service needs to be registered at the central service registry. This is done by creating an image of the service using a Dockerfile and then uploading it to the registry. Harbor is an open-source registry and used as the central service repository, and GitLab is used for version control and to automate the image creation step with the internal continuous integration and deployment (CI/CD) pipeline. The last step in the scenario definition phase is to define a docker-compose file that contains the default configuration for service deployment. It serves as a template that is customized for SMP deployment.

To simplify service development and deployment and thus reducing the time from prototypical service implementation to production, template projects in GitLab are utilized. The templates come with predefined configurations and settings for various tasks such as code formatting, Python application packaging with semantic versioning, Docker image creation, and standard CI/CD pipelines, enabling automated testing and image creation. Moreover, a default docker-compose file is available for service configuration which can be customized by the service developer. The purpose of providing a project template with these default configurations is that the service developer can focus on the task implementation and does not have to deal with those processes, which can be challenging and time-consuming, as indicated by Reis *et. al.* [122].

SMP configuration and deployment The scenario mining process has been defined based on a scenario definition, which is then decomposed into individual tasks. Each task is semi-automatically wrapped in a containerized service, with each service providing a default configuration as a docker-compose file.

The next step in the overall process is to configure and deploy the SMP and thus realize the scenario mining process using the docker-compose files of the services. Figure 6.8 illustrates both phases using the representation of the SMP from Figure 6.3. The scenario description and the scenario mining process derived from it serve as the basis for configuring the SMP. For that purpose, a docker-compose file is created that contains the definitions of the required services, *i. e.*, the services are selected that are required to realize the scenario mining process. To enable service collaboration, it must be specified how they communicate with each other via the topics. That is, for every service the topic it subscribes to and, optionally the topic(s) to publish messages to must be specified. Although every service subscribes to a default topic, this can be customized by overwriting the default values in the docker-compose file. Additionally, to specify how traffic data flows into the SMP, the data input connectors must be configured according to the use case. Finally, to save service results along the scenario mining process, the output connector(s) must be specified. They typically subscribe to topics, process the results and persistently save them.

Note that docker-compose is employed to deploy the SMP instance either on a single machine or as a distributed system across multiple machines. For container orchestration, however, Kubernetes is used. Therefore, during SMP deployment phase, it is necessary to convert the docker-compose file to Kubernetes configuration files.[4] For that purpose, this work utilizes `Kompose`[5], an open-source tool that translates docker-compose files into Kubernetes resource configuration files. After conversion, the SMP can be deployed by applying the converted configuration files. The last phase after SMP deployment is also dependent on the specific use case, in which the extracted scenario are analyzed. This is usually achieved by using components of Scenimini's application layers, which are utilizing the TAP-API for data access and management.

6.3 Traffic Processing API

The Scenario mining pipeline (SMP) introduced in the last section is a distributed system with a service-oriented architecture (SOA) that is utilized to find and extract scenarios of interest from traffic data. The SMP is composed of several services, each wrapping a particular task of the scenario mining process. Given a scenario description, that scenario mining process must be defined and service developers must implement missing services. The TAP-API offers standardized interfaces for service developers, but also for application developer to access the data tier of Scenimini. For that purpose, the TAP-API is composed of three components providing access to the data tier at three levels of abstraction and for different use cases. An overview of the TAP-API is illustrated in Figure 6.9 following a hierarchical model with arrows indicating information topics between the components.

[4] This step is necessary only if the SMP is to be deployed as a distributed system on a Kubernetes-managed cluster. If the SMP is to run on a single host and no container orchestration is required, it can be run with docker-compose.

[5] `https://kompose.io/`

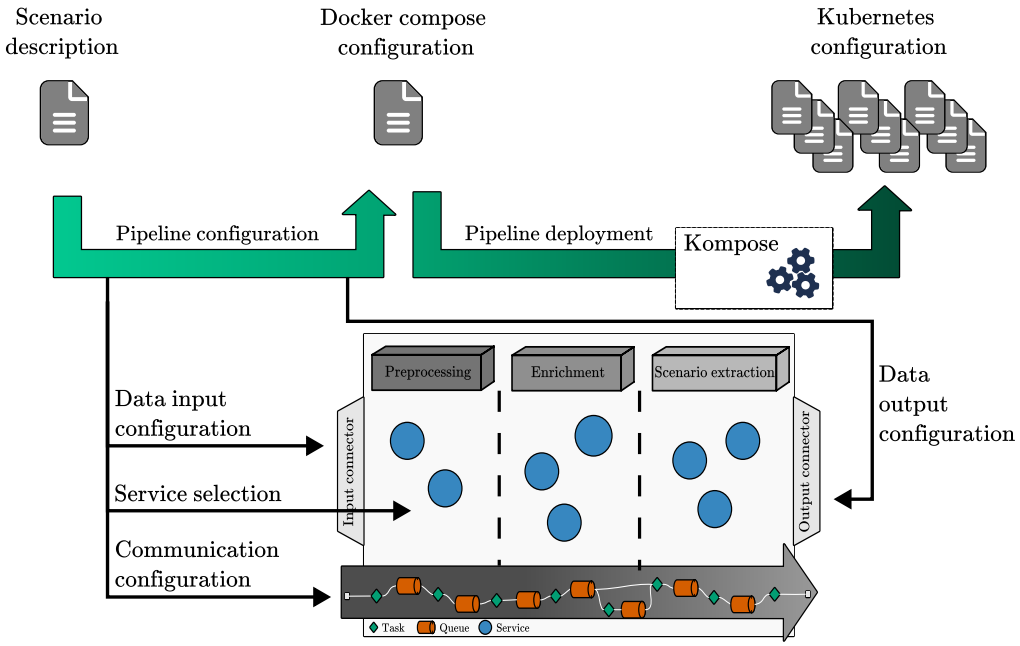


Figure 6.8: The process from a scenario description to a configured and deployed scenario mining pipeline instance.

The first component ScenORM of TAP-API is introduced in Section 6.3.1 and enables service developers to manage extracted scenarios and traffic data in a RDB via Python while adhering to an object-oriented paradigm. To achieve this, an Object-relational Mapping (ORM) is utilized to link Python objects with tables in the RDB, while also concealing Structured Query Language (SQL) intricacies from the developer.

The second component of TAP-API focuses on the efficient and high-performance processing of traffic data in Python. It is a Traffic situation analysis and interpretation (TASI) library designed for traffic data processing with adaptors to support various data sources. It employs ScenORM to manage traffic data via ORM models and provides access to the *GeoServer* for, *e.g.*, visualization or analysis. Section 6.3.2 will introduce TASI, while focusing on the data models used to internally represent trajectory traffic data and on application domains in which TASI is employed.

The TAP-API's third component traffic and scenario API (TraSceAPI) builds on top of the previous two. It is an API adhering to the Representational State Transfer (REST) programming paradigm developed by Fielding [123]. Its main purpose in a distributed system of microservices is, among other things, to decouple components from each other while at the same time providing access via standardized interfaces. [123] In Scenimini, TraSceAPI decouples applications and services from the data tier. The SMP services and Scenimini applications can use the API to access the data tier via Hypertext Transfer Protocol (HTTP) or secured HTTP (HTTPS) requests. In the following, a brief overview of the three components is given.

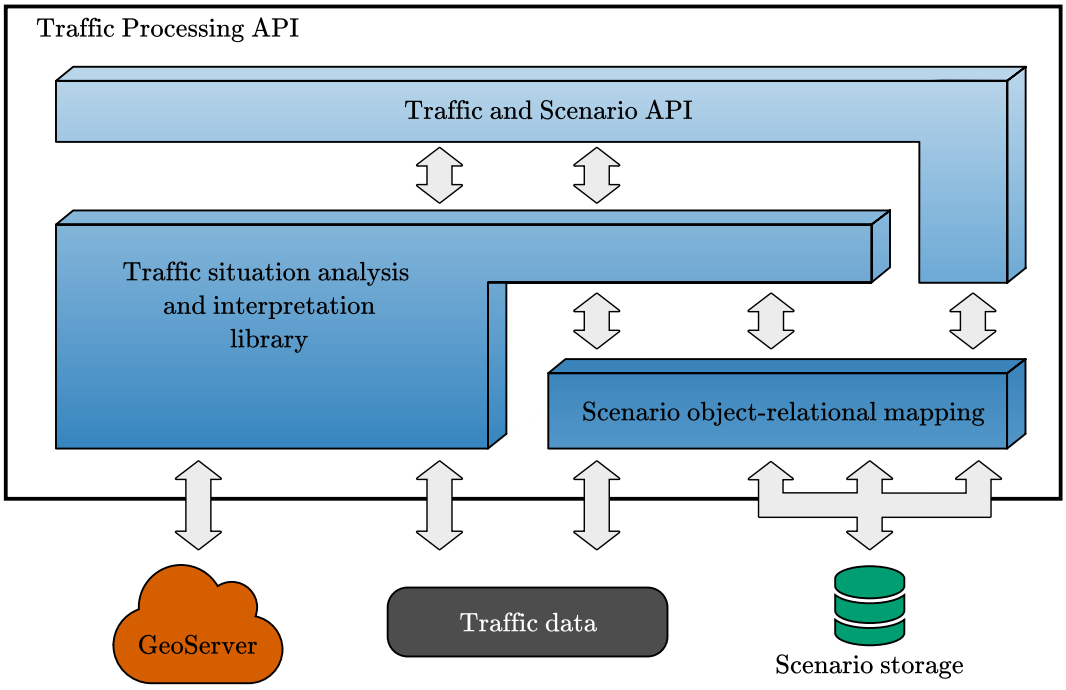


Figure 6.9: The Traffic Processing API provides access to the data tier at three levels of abstraction.

6.3.1 Scenario object-relational mapping

The first component of the TAP-API is the Scenario object-relational mapping (ScenORM) that enables to persistently store scenario-related details extracted from traffic data by the SMP in a relational database (RDB) while at the same time following a object-oriented programming paradigm. To achieve this, an Object-relational Mapping (ORM) is typically employed that abstracts the database access and facilitate code clarity, clean code and extensibility.

In the last decades, ORMs have been extensively studied and used to combine object-oriented programming with RDBs. The primary issue is the challenge of mapping an object-oriented programming language view onto a relational database view, commonly referred to as the *impedance mismatch*. For additional information on this problem and an overview of potential solutions, please refer to the survey conducted by Torres *et. al.* [124]

This work employs SQLAlchemy[6] to design ScenORM in Python. Since the SMP can be utilized to mine arbitrary scenarios in traffic data, ScenORM should support representing different variants of scenarios. To achieve this, inheritance is employed in Python which also enables sharing common attributes across scenarios. However, RDBs usually do not support this feature. Instead, ORM tools typically use three patterns for *inheritance mapping* [124][125].

[6] `https://www.sqlalchemy.org/`

The inheritance graph illustrated in Figure 6.10 (*a*) is exemplarily realized in Figure 6.10 (*b* − *d*) using the three patterns. Figure 6.10 (*b*) shows the *Single Table Inheritance* pattern where all attributes of the classes `A,B,C` are combined in a single table with an additional column to denote the actual class. The second inheritance pattern *Class table* is shown in Figure 6.10 (*c*) where every class is mapped onto one table. The inheritance hierarchy is represented via associations between the parent and child classes. Due to this, every table contains only the class attributes. The third inheritance pattern *Concrete table* in Figure 6.10 (*d*) is a combination of the previous two patterns. Every class is mapped onto one table as in the *Class table* pattern, but contains all attributes along the inheritance path. Choosing the mapping strategy to represent inheritance is crucial since it "*affects behavior, performance, and design limitations*" [124].

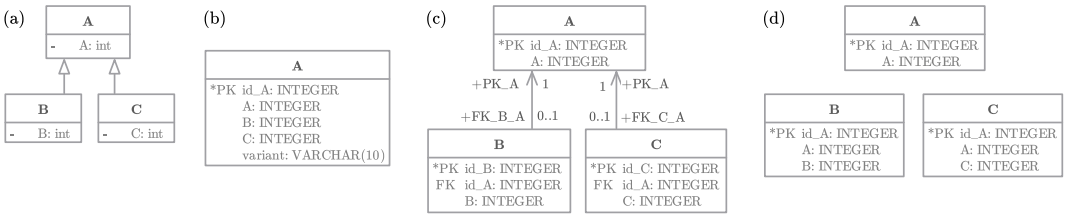


Figure 6.10: The three patterns (b - d) typically used for inheritance mapping by an ORM for the example illustrated in *a)*. [124] *b)* Represent all attributes in a single class *c)* Map each class to one table and add associations to the parent class. *d)* Map each class to one table with every table containing all attributes of the inheritance path.

For the excerpt of ScenORM illustrated in Figure 6.11, a strategy is chosen with a variant of the *Concrete table* where the superclasses are abstract, but all scenario variants and other entities are mapped onto one table. The advantages of this approach are that class instances can be retrieved from the RDB without joining along the inheritance path, which is required by the `Class table` pattern. Additionally, attributes can be shared across scenarios through the use of the abstract superclasses. ScenORM can also be easily extended with additional scenario definitions without altering the table structure, as would be necessary with the `Single table` pattern. This is a crucial feature since it allows for dynamic extension of the scenario definitions of the SMP and their representation in the RDB.

Figure 6.11 illustrates an excerpt of the ScenORM architecture to represent crossing scenarios, which were extracted in Chapter 5. The main entities that are represented as tables in the RDB are the `Scene`, `Pose`, `RoadUser`, `EgoParticipant`, `ChallengerParticipant` and the `CrossingScenario`. The `Scene` is used as an anchor to describe the environment for a particular point in time. The `Pose` is a subset of the scene and describes the state of a traffic participant for a specific scene. The `RoadUser` describes a traffic participant over the full course of the lifetime without the trajectory. A `RoadUser` can participate in various scenarios as either the *ego* or *challenger* participant. Extracted crossing scenarios are represented by the `CrossingScenario`. Note that all other classes in Figure 6.11 are abstract classes, typically indicated by the `Mixin` postfix in Python. Those `Mixins` are used to share common attributes across multiples tables on the RDB side and operations on the Python

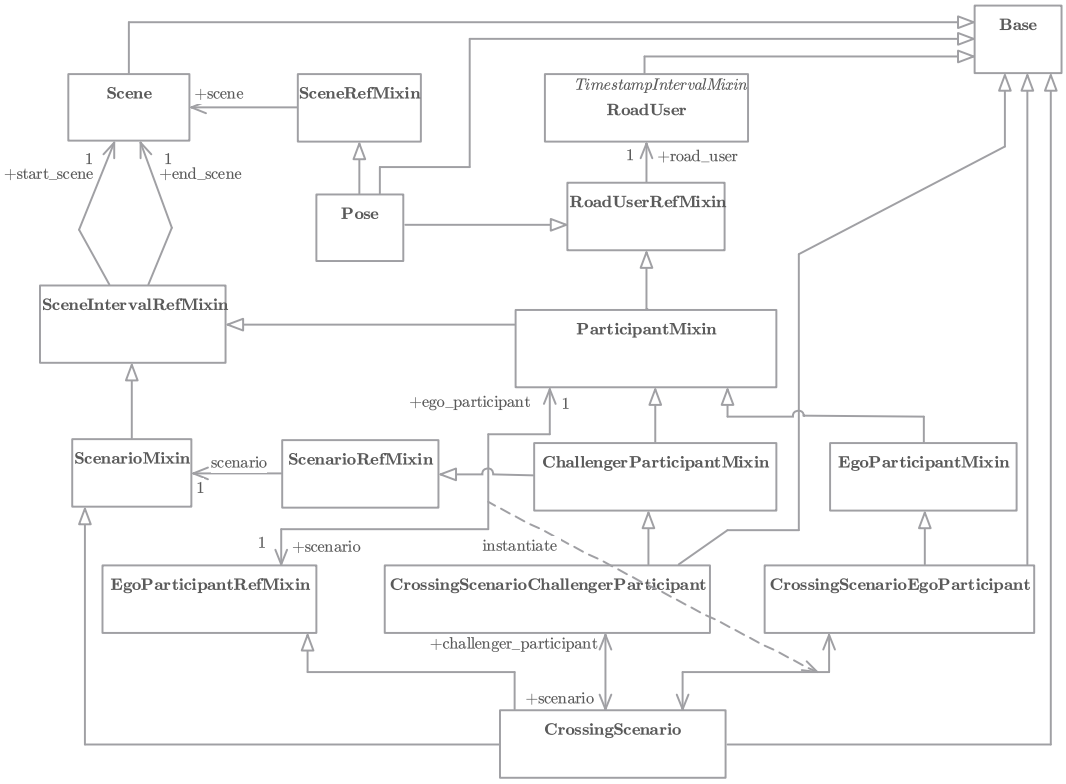


Figure 6.11: An excerpt of the ORM for the scenario-driven database architecture using the crossing scenario for illustration purpose.

side. [126] For instance, the `ScenarioMixin` contains general scenario definitions, which are inherited by the `CrossingScenario` or any other scenario variant. The `ScenarioMixin` is thus the abstract superclass and is not mapped onto a table. The same applies to the `Participant` which shared attributes are defined in the `ParticipantMixin`. The class is specialized by the child classes `EgoParticipantMixin` and `ChallengerParticipantMixin` to represent the *ego* and *challenger* participant. Note that both classes are also abstract classes and are thus not mapped onto a table. Since every scenario variant is mapped onto one table, there is also a class representing the *ego* and *challenger* participant in Figure 6.11 for that particular scenario (`CrossingScenarioEgoParticipant`, `CrossingScenarioChallengerParticipant`). Adding a new scenario to ScenORM is straightforward and only requires to create new classes for the scenario and both the ego and challenger participants using the mixins.

In SQLAlchemy all classes that should be mapped onto one table in the RDB inherit from the `Base` class. This is exemplified in Figure 6.11 using the Oracle database management system (DBMS). Since the SMP uses ScenORM to access the RDB and should be used in different use cases, e.g. to process live data or campaigns, a custom base class was created. It allows to specify a namespace all tables belong to, which is realized by dynamically adding the namespace to every table name. This is illustrated in Figure 6.12 showing the tables

in the RDB the classes of ScenORM are mapped onto and the associations between them. Since `ScenORM` is the default namespace, every table has this prefix. This strategy not only ensures that data from different use cases are physically separated in the RDB. It also allows reflecting versioning of ScenORM in the RDB by adding version information to the namespace.

Thus far, it was illustrated how to represent class inheritance of ScenORM's python side on the RDB side. Instead of sharing attributes between classes, this approach is also used to easily establish associations between entities. In fact, all mixins in Figure 6.11 with the term `Ref` are used to create associations between entities that are shown in Figure 6.12. For instance, the `CrossingScenario` entity allows to navigate to the `CrossingScenarioEgoParticipant` instance via the `ego_participant` property. This is realized by the `EgoParticipantRefMixin` that defines a relationship in ScenORM from a scenario variant to a class that inherits from the `EgoParticipantMixin`.

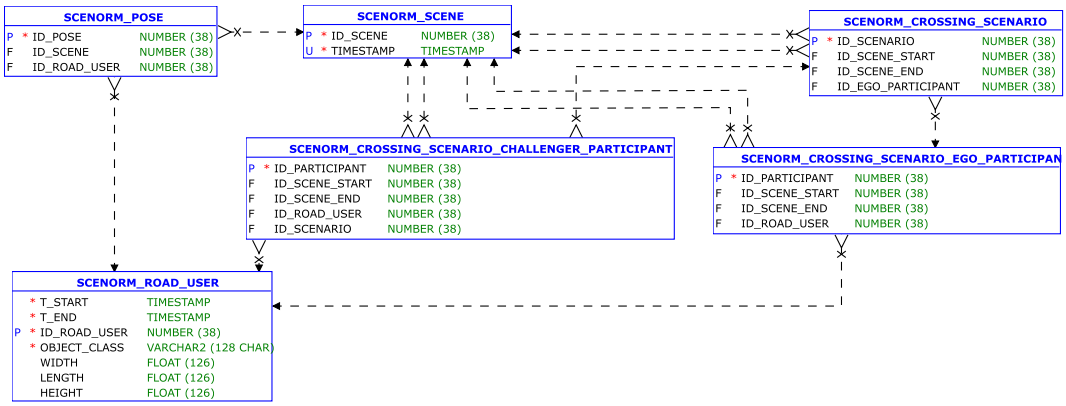


Figure 6.12: The table instances created based on the `ScenORM` architecture for the crossing scenario.

6.3.2 Traffic situation analysis and interpretation library

The presented ScenORM in the last section is the first component of the SMP and allows to manage scenarios in a RDB. The scenario mining process realized by the SMP typically includes processing and analyzing the trajectories of traffic participants or environmental effects. For this purpose, the SMP's second component is the Open-Source library TASI, which provides high-performance data structures and data analysis tools to manage and process traffic data in Python. [127] In the following, an overview is given of the data models used in TASI for trajectory data representation, which serve as the foundation for traffic analysis tools and briefly highlight some application domains.

In TASI, a road user's trajectory is a series of state measurements. Each state contains several attributes, but at least the road user's location in UTM and dimension in longitudinal and lateral direction (in local vehicle coordinate space). This enables the support of measurements from various sources that typically do not have a common reference position. For example, position measurements from a vehicle's self-localization system may

be located somewhere within the vehicle and depend on the system's location and/or the antenna. The same is true for traffic infrastructure measurements, where cameras may locate traffic participants and the reference position may even change in the local vehicle coordinate space. The post-processing of the position measurements to a common reference position automatically take place by TASI. That is, these base attributes are utilized to estimate the road user's dimension, velocity (in UTM), speed, acceleration (in UTM), and heading in SI units, if they are missing. In addition, various reference positions of the road user are provided for analysis purposes, as illustrated in Figure 6.13 with the gray triangle denoting the road user's heading. The exterior of the road user for a two-dimensional representation is defined by either the four or eight corner points for a three-dimensional representation using the road user's height. The `location` attribute is the initial reference position, while `origin` is the position used for all calculations that involve position information. By default, `origin` represents the front center position of the road user, which can be customized to the specific needs.

Thus far, it was shown how traffic participants are represented within TASI given a minimum set of measurements. We will now follow with the data structures that are used in TASI to represent trajectory-based traffic data. The data models of TASI are based on the structures defined in the two widely used Python libraries `pandas`[7] and `GeoPandas`[8], while the representation format is used with respect to the use case at hand. Hence, this work follows a different approach than the `trajdata` framework [128], which main purpose is to provide access to existing trajectory datasets using a custom internal representation format that can be converted to a PyTorch [129] `Dataset` for further processing. `pandas` is chosen as the base for TASI, since it is widely used and thus provides interfaces to various other libraries such as `PyTorch` and `Matplotlib` [130] and supports various data formats such as comma-separated values (CSV) and Apache Parquet [131]. Before going deeper

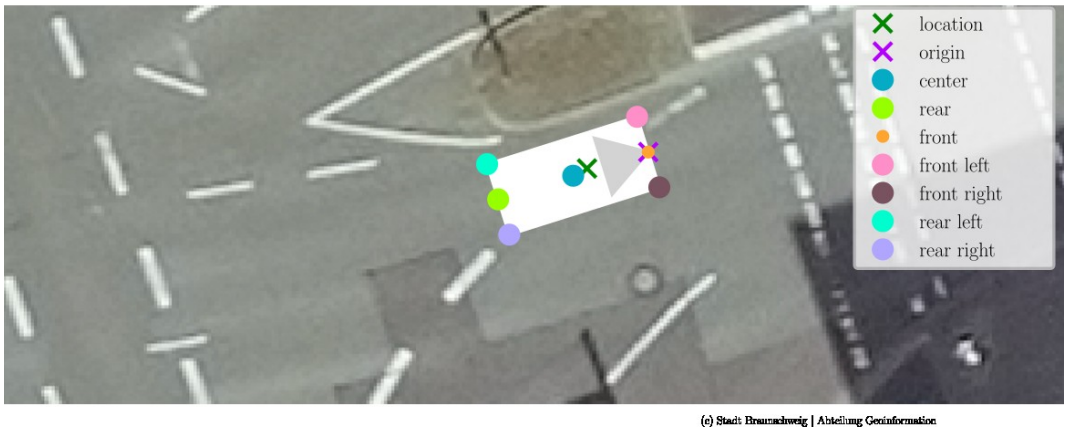


Figure 6.13: The two-dimensional representation of a traffic participant's pose with all estimated positions.

[7] `http://pandas.pydata.org/`
[8] `https://geopandas.org/en/stable/`

into the TASI data models, it will be briefly described how data is managed in pandas and GeoPandas and follow-up on how TASI is connected to the libraries.

Pandas data structures The python library pandas [132] is the de facto standard in Python to work with tabular-formatted structured (or relational) data. To represent data in the tabular-styled format, pandas uses two primary models to hold information, the Series and DataFrame (see Figure 6.14). The DataFrame is a two-dimensional structure, which is composed of multiple Series. Every Series represents a column of the DataFrame and contains homogeneous-typed information. By stacking Series, heterogeneous data can be represented in the DataFrame.

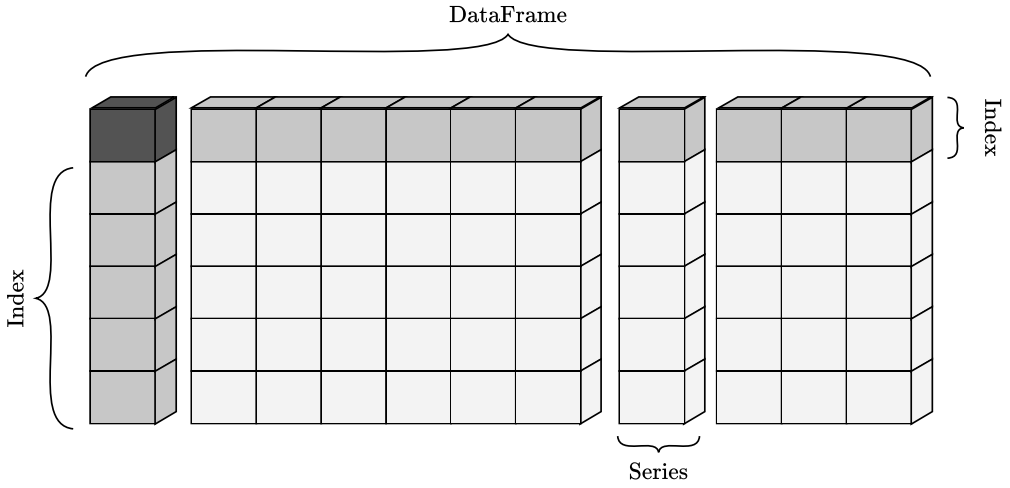


Figure 6.14: The pandas data structures to represent data in a tabular-styled format.

A structured data set usually contains data whose values are given a specific label. In pandas, the Index structure contains those labels. They also allow accessing specific information of the DataFrame. As illustrated in Figure 6.14 the Index structure describes the content for a particular row and column, while the DataFrame's property index refers to the Index that labels row-wise and the property columns to the Index that labels column-wise. The Index also allows to hierarchically index columns and rows, *e.g.*, to represent values with a tuple of labels.

GeoPandas data structures The second library TASI uses for trajectory data representation is GeoPandas [113], which itself is also based on pandas and extends its capabilities to support geographic data management in Python. For that purpose, it supports geometric objects of Shapely[9]. Thus, it allows the analysis and manipulation of such geometric objects in an efficient way since Shapely employs the widely used GEOS library [133]. The two primary structures GeoSeries and GeoDataFrame in GeoPandas are specializations of the Series and DataFrame of pandas (see Figure 6.15). The GeoSeries hold geometric objects

[9] https://shapely.readthedocs.io/en/latest/

which are typically `Shapely` geometries and it provides access to geometric operations. The `GeoDataFrame` is used to represent a collection of `GeoSeries` and `Series`. Spatial operations are, however, only applied to a specific `GeoSeries` of the `GeoDataFrame`, which is accessible via the `geometry` attribute indicated in Figure 6.15 as the *geometry* column. The other columns allow to add additional information related to the geometric objects or even other geometric objects.

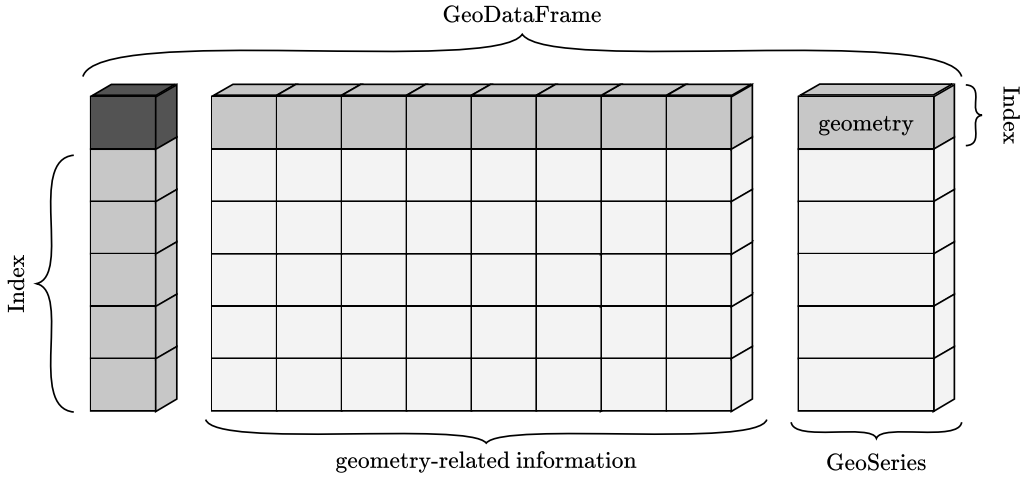


Figure 6.15: The `GeoPandas` data structures to manage geometric data in a tabular format based on `pandas`.

TASI data structures Based on the data structures of `pandas`, `TASI` defines three primary data models for trajectory data representation as illustrated in Figure 6.16. The first entity in TASI is the `Pose` which, analogous to the `Pose` in ScenORM, is the traffic participant's representation at a specific time. This analogy is by design, as it is used to couple TASI with ScenORM and thus manage trajectory data in an RDB via an ORM.

The second entity `Trajectory` is a composition of `Poses`. That is, a collection of at least two `Poses` of the same traffic participant is a `Trajectory`. It enables the representation of the course of traffic participant's trajectory over time. If a collection contains at least two `Trajectory` entities, it is a `Dataset`. This is the third entity that also provides methods for data import from files or databases, as well as export into various formats such as CSV, Parquet, or even OpenSCENARIO.

The `PoseCollection` illustrated in Figure 6.16 is a utility base class to, *e.g.* adapt the default indexing behavior of `pandas` and define common operations for both the `Dataset` and `Trajectory`. Moreover, it is a fallback representation if a collection contains, for instance, two `Poses` of two different traffic participants. Note that all three models inherit from the `DataFrame`.

The major advantages of this approach are that all the operations provided by `pandas` are available and can be extended by functions specific to the traffic domain, *e.g.*, trajec-

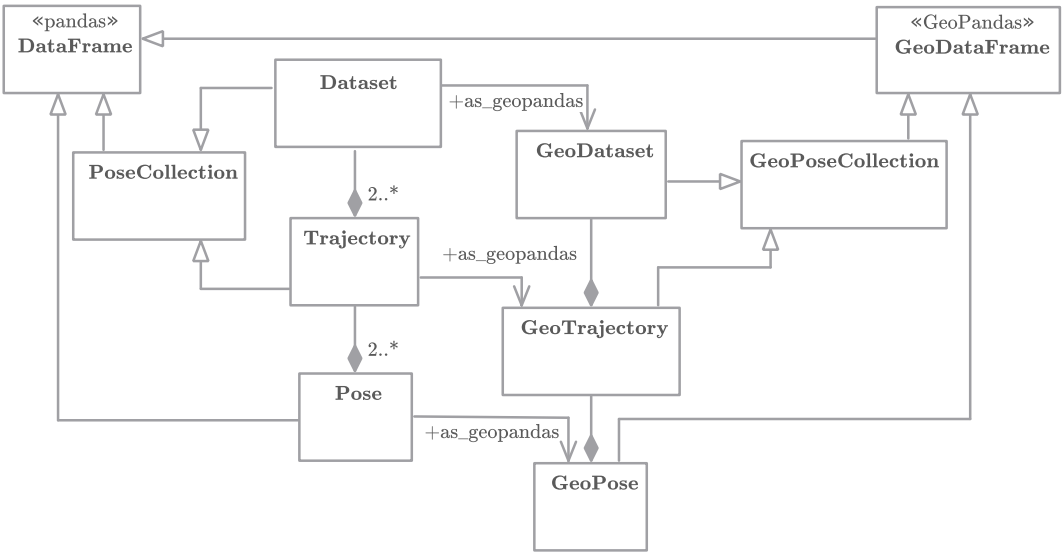


Figure 6.16: The TASI data models to manage trajectory data from traffic participants using `pandas` and `GeoPandas`.

tory smoothing, scenario traffic data export or estimation of Surrogate Measures of Safety (SMoS). For example, in Chapter 5, the extraction of crossing scenarios requires to estimate the PET between traffic participants. A service developer can easily achieve this since the method is available at the `Trajectory` class. It allows the estimation of the PET between a trajectory and other trajectories.

In `GeoPandas`, the `DataFrame` and `Series` are specialized for the purpose of representing geometric objects. To represent trajectory data as illustrated in Figure 6.16, however, a slightly different approach is followed than in `GeoPandas`. Since all three models inherit from a `DataFrame`, the `Index` is utilized to differentiate between the models.

The tabular representation of the TASI models based on `pandas` is illustrated in Figure 6.17. The `Dataset` represents the overall collection. For the row `Index`, a hierarchical index is used which is composed of the traffic participant's `id` and a `timestamp`. Hence, a row of the `Dataset` represents the state of a traffic participant for a specific time, *i. e.*, the traffic participant's `Pose`. This is exemplarily indicated in Figure 6.17 with the row $(t_2, 0)$ where the traffic participant's pose with the id 0 at the time t_2 is selected.

If multiple rows of the `Dataset` are selected that cover a time interval but have the same `id`, the `Trajectory` of that traffic participant for that time interval is selected. This is, for instance, the case for the rows with an index of $(t_0, 0)$, $(t_1, 0)$, $(t_2, 0)$. Since the rows have the same `id` but different `timestamps`, they are represented as a `Trajectory`.

A hierarchical index is also utilized for the column `Index`, since the columns of the `Dataset` contain the `Pose`'s `attributes`. This is due to the fact that attributes can be nested and the proper way to represent them in a tabular format is to place them on different levels of the `Index`. For instance, the dimension of a traffic participant can be specified by its

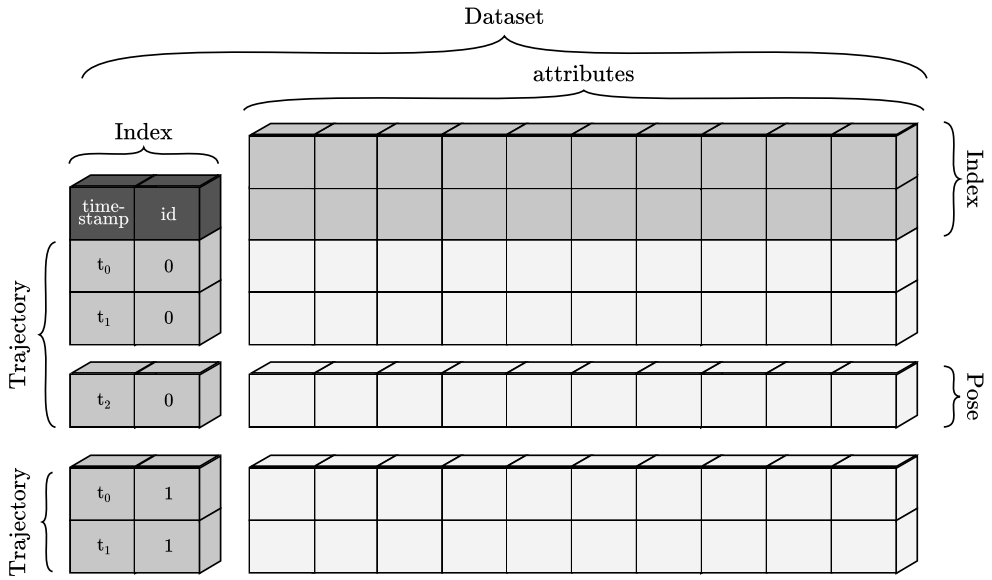


Figure 6.17: The TASI primary data models to represent trajectory data based on `pandas` using a hierarchical index for model differentiation and pose attributes.

width, height and length. The first level of the `Index` would be the `dimension` and the second level would contain the width, height and length.

For every of the three data models presented thus far, there is a specialization in `TASI` to represent the position information of the trajectory data as geometric objects. For that purpose, three models are defined that are all based on the `GeoDataFrame` of `GeoPandas`. The `GeoPose` contains the information of the `Pose`, but represents the position as a geometric object. Analogous to the `Trajectory`, an `ObjectTrajectory` is a collection of multiple `GeoPoses` and a `GeoDataset` is a collection of multiple `GeoTrajectories`.

The tabular representation of the TASI models based on `GeoPandas` are illustrated in Figure 6.18. The overall collection is a `GeoDataset`, and analogous to the `Dataset`, the `Index` is composed of the `timestamp` and the traffic participant's `id`. The `geometry` column specifies the traffic participant's position as geometric object. All other columns are attributes describing the traffic participant's pose. The differentiation between the three data models takes places analogously to the `pandas`-based models with respect to the `Index`.

The TASI models that are based on pandas can be converted to their counterparts that are based on `GeoPandas`. That is, there is a one-to-one mapping from the `pandas`-based models to the `GeoPandas`-based models. This is also indicated in Figure 6.16 with the `as_geopandas` navigation attribute.

Other than the `pandas`-based models, the `GeoPandas`-based models are typically used for visualization purposes or when geometric operations need to be applied. An example use case is the analysis of traffic participants in a specific region of interest, *e.g.*, an on-ramp section of a motorway or a specific intersection of a road network. For that purpose, only the poses within the region of interest should be selected. If the region is available in OpenDRIVE, and therefore as a `GeoDataFrame` via [114], the `GeoPoses` that are within

that region can be efficiently identified using `GeoPandas`' functions. In some cases, time information may be unnecessary and geometric operations should be applied on the trajectory level. For this purpose, the `Trajectory` can also be converted to a `GeoDataFrame` by aggregating all poses and thus representing it with a single geometric object. That is, trajectories are converted to a representation as `LineString` that allows to apply geospatial operations.

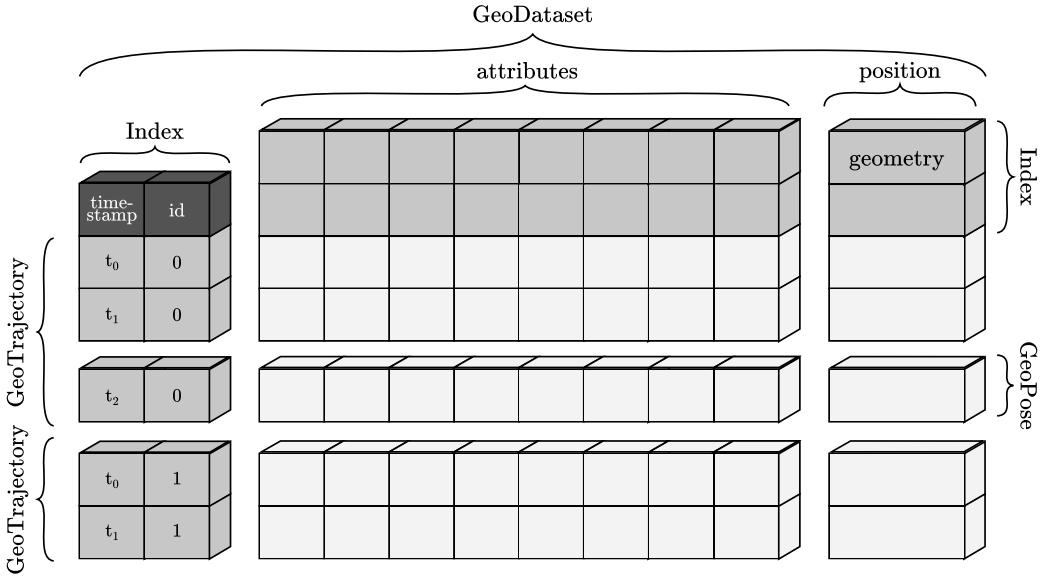


Figure 6.18: The TASI data models based on `GeoPandas` to represent trajectory data with selected position information as geometric objects.

Application domains The TASI data structures enables to represent trajectory data of traffic participants based on the concepts of datasets, trajectories and poses using either `pandas` or `GeoPandas`. Moreover, this representation also enables to process trajectory data. Based on the introduced data structures, TASI provides a tool set of methods for that purpose. We will now focus on some tools provided byTASI and where it is used within Scenimini, while highlighting specific use cases.

One of the modules provided by TASI contains methods for the estimation of SMoS through standardized implementations. These methods are particularly relevant for situation analysis in terms of traffic safety [134][66]. For instance, the data structures allow to easily estimate the PET or TTC between traffic participants which are provided by dedicated services of the SMP. For instance, the PET was used in Chapter 5 for filtering crossing scenarios according to a maximum PET or in a study of Da Silva *et al.* [135] for the same purpose, but with focus on the interaction between bicyclists and motorized traffic.

Scenario descriptions usually specify the road environment and traffic data is typically filtered by the driving direction of traffic participants. Due to that, TASI contains a module that enables to associate traffic participants with roads. In fact, the module provides

Figure 6.19: An example application of TASI for route estimation without a digital representation of the road network. The reference line for each routing direction is estimated based on trajectories from a subset of the overall data set.

several methods for this task. These range from approaches based on map matching as shown in Section 3.4, using DAGMaR and a digital representation of the road network, to methods that use hand-crafted polygons and that allow for traffic filtering without a digital representation of the road network.

Figure 6.19 presents an example of route estimation using map matching, without a high-definition digital representation of the road network. This was demonstrated in a measurement campaign during the VRIEDRICH[10] project. The project aims to derive and discuss measures to improve road safety in the rural areas based on practical findings of local traffic conditions. To achieve this, it is necessary to identify the typical driving routes of all traffic participants. Due to the unavailability of a high-definition digital representation of the road network, a subset of the overall dataset of trajectories was utilized to estimate the reference trajectories for each primary driving route and other relevant routes (such as the path of buses via the bus stop) at the intersection. The reference trajectories serve as the digital representation of the road network, enabling the utilization of the map matching based approach as shown in Section 3.4.

Figure 6.20 illustrates another example for route estimation for route-based traffic filtering using hand-crafted polygons. The approach is inspired by using physical inductive loops for vehicle detection on intersections [136], with the polygons serving as virtual loops. This is a typical approach to filter a dataset by driving direction [137], for traffic density estimation [138][139], or for triggering specific events [140]. The left panel of Figure 6.20 shows the trajectories of the dataset acquired by the DLR AIM Research Intersection. The aim in this example is to identify only the traffic participants entering the

[10] https://verkehrsforschung.dlr.de/de/projekte/vriedrich-verkehrsbeobachtung-und-analyse-im-laendlichen-raum-im-kontext-der

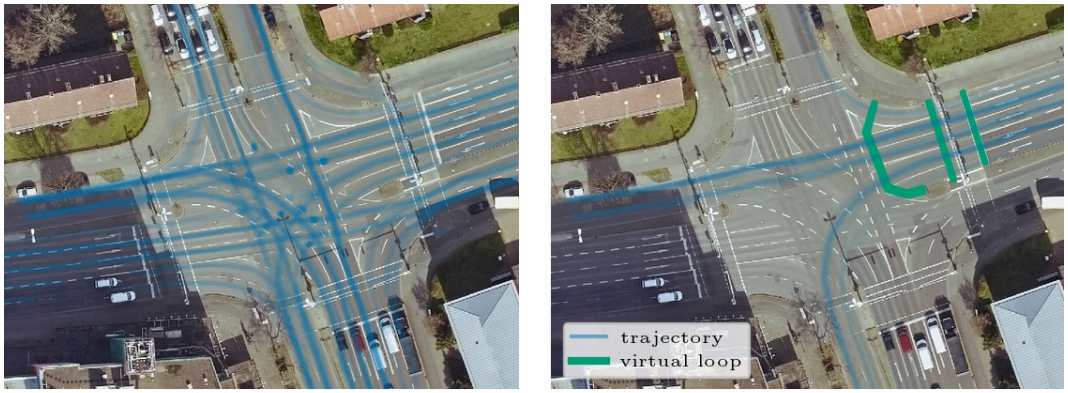


Figure 6.20: An example application of TASI for route estimation based on hand-crafted polygons serving as virtual loops for road user selection. *Left:* The road user trajectories of the dataset. `Right`: The virtual loops for selection and the trajectories of the selected road users.

intersection from east. This can be accomplished with the virtual loops illustrated in the right panel of Figure 6.20, which are hand-drawn via bokeh[11] and can be exported as GeoJSON files. Since *GeoPandas* support the file format, the trajectories intersecting with the virtual loops can be easily found using the TASI `GeoDataset`. The trajectories of the road users after filtering are also illustrated in Figure 6.20, showing only those trajectories that geometrically intersect with the virtual loops.

Trajectories of traffic participants can come in different formats as this typically depends on the system that is used to obtain them. However, since trajectories are multivariate time-series, they can be represented with the TASI data structures. This makes it possible to use standardized implementation in TASI for different datasets and use cases. A module of TASI is dedicated to this and provides tools for data exchange particularly of trajectory data, which are used in the input and output connectors of the SMP.

One use case is the management of on-board sensor data, together with additional sensors, for system analysis as part of the FASva project[12]. During the project, the VuT (as illustrated in Figure 6.21) was used to collect measurement data using the Automotive Data and Time-triggered Framework (ADTF)[13]. ADTF allows to serialize sensor data into, so-called, *DAT* files and provides tools for file management. The VuT state estimates are exported to CSV formatted files through the ADTF C++ API. An interface in TASI loads these CSV files that contain various information such as the position in GPS and speed using *pandas* and converts them to a TASI trajectory. This enables the use of, for instance, the route estimation method presented in Section 3.4 with trajectories acquired from the AIM Research Intersection or the Testbed Lower Saxony, even though it was initially proposed in [89] using trajectories of the VuT from Figure 6.21.

[11] `https://bokeh.org/`
[12] `https://www.hs-emden-leer.de/studierende/fachbereiche/technik/projekte/fasva`
[13] `AutomotiveDataandTime-TriggeredFramework`

Figure 6.21: The measurement vehicle used within the FASva project with the measurement system located in the trunk of the vehicle (right panel). © Hochschule Emden/Leer

Another use case involves the qualitative analysis of a test bed's performance such as the Testbed Lower Saxony[14] using a vehicle equipped with a high-precision self-localization system. The reference vehicle's measurements serve as the ground truth and allow for the estimation of the test bed's performance. However, to achieve this, the trajectories provided by both systems need to be homogenized by converting both information sources to a TASI-compatible format. In this use case, the reference vehicle illustrated in the left panel of Figure 6.22 uses a system that is based on the Robot Operating System (ROS) and equipped with, among other things, a GNSS-based inertial navigation system (see right panel of Figure 6.22). The information collected by sensors in ROS is published as messages in ROS topics. Similar to ADTF, they can be stored persistently for offline processing, but in this case, as *bag* files. TASI provides an interface based on the *rosbag*[15] library to convert the *bag* files. Every state estimate including position information in GPS and speed is converted into a TASI `Pose`, while the vehicle dimension need to be specified on import. A `bag` file typically contains multiple state estimates, which are aggregated into a TASI `Trajectory`. This enables the development of standardized tools for this use case, such as trajectory interpolation and comparison, which can be reused in other environments.

The last application domain worth noting, since it of high interest in the academic and industrial domain, is the simulation-based testing of automated driving function (ADF). Traffic data from the real world provide scenarios that can be used in simulation for in-depth analysis of the scenario participants. The OpenSCENARIO standard by the Association for Standardization of Automation and Measuring Systems (ASAM) defines an exchange format for the dynamic world context of such scenarios and thus enables to represent the scenarios extracted from the real world in the simulated world. TASI provides an interface on the `Dataset` level that is based on *scenariogeneration* library[16] and enables to represent the trajectories in an OpenSCENARIO file, replicating the traffic participants

[14] `https://verkehrsforschung.dlr.de/en/projects/test-bed-lower-saxony-automated-and-connected-mobility`

[15] `http://wiki.ros.org/rosbag`

[16] `https://github.com/pyoscx/scenariogeneration`

Figure 6.22: The FASCar of the DLR (left panel) as a reference vehicle for qualitative analysis of test beds using TASI, with the measurement system located in the trunk of the FASCar (right panel). © DLR

behavior in simulation. Figure 6.23 shows an example situation with traffic participants of the DLR AIM Research Intersection. The trajectories are clipped using the OpenDRIVE digital map to the inner-intersection area for demonstration purpose only. The left panel illustrates the situation visualized using TASI. The scenario is converted to the OpenSCENARIO format and replayed in ESMini as shown in the right panel of Figure 6.23.

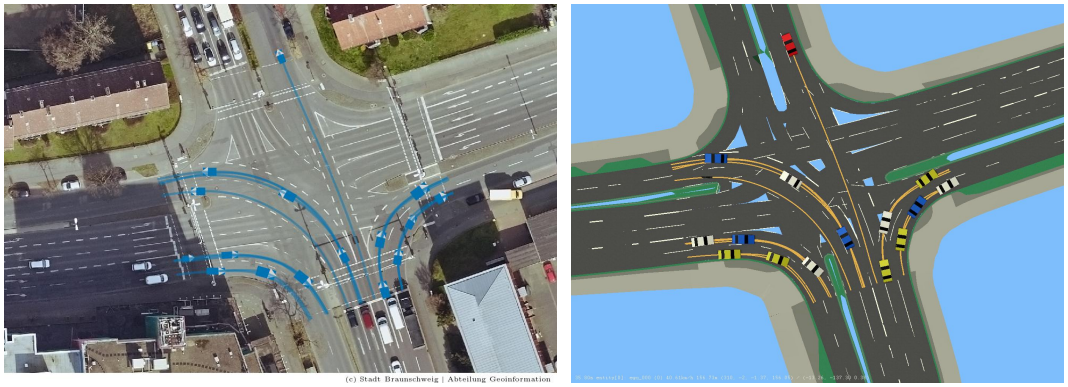


Figure 6.23: TASI allows the replication of real-world scenarios in simulation. *Left:* A real-world scenario visualized with TASI. *Right:* The scenario replayed in ESMini by converting the TASI dataset into the OpenSCENARIO format.

6.3.3 Traffic and Scenario API

The traffic and scenario API (TraSceAPI) is the third component of the TAP-API and builds upon the previous two components. It main purpose is to provide SMP services and applications access to traffic data and the extracted scenario via standardized interfaces.

The RESTful API is also designed as a service running in the Kubernetes Cluster. This allows to scale the RESTful API, and because it is orchestrated by Kubernetes, ensure that

instances are kept up online in case of temporary failures. The API is implemented in Python using the Sanic API[17] library. This choice was made because it provides an API description in the OpenAPI specification format[18], a simply interface to create websocket servers for video streaming of traffic infrastructure cameras and allows easy deployment.

6.4 Applications and discussion

The Scenimini platform is used in several projects and environment for scenario extraction and analysis. In the following, an overview of different use cases and specific components of Scenimini is given.

6.4.1 AIM Research Intersection

The DLR Research Intersection is part of the Test field AIM and enables to study traffic behavior on a complex urban intersection in Braunschweig, Germany. [42] To achieve this, the intersection is equipped with 14 stereo-camera systems, among others, providing a three-dimensional representation of traffic participants and their trajectories. The Research intersection is used in several projects as illustrated by Knacke-Langhorst [141] with Scenimini enabling researchers to focus on the scenario-driven data analysis by providing the necessary tools. In the following, a brief overview of the applications of Scenimini or some of its components using traffic data of the DLR Research Intersection is given.

The Institute of Transportation Systems at DLR participated in the Validation and Verification Methods (VVM)[19] research project to provide and analyze scenarios collected at the Research Intersection, among others. In a first study [114], Scenimini was used to associate traffic participants to typical routes to study the behavior of traffic participant crossing the intersection, while focusing on scenarios with traffic participants performing a u-turn maneuver. The analysis considers only those parts of the trajectories of traffic participants that are within the intersection. Additionally, trajectories are projected onto the Frenét coordinate system (see Section 2.1.2), which is defined by the u-turn lane, as shown in the bottom panels of Figure 6.24. This division allows for a geometric analysis of the trajectory, enabling the examination of the behavior of traffic participants when entering, crossing and leaving the intersection.

Figure 6.24 shows results of this study. The top row displays the distribution of the mean and maximum positional deviation in each of the three phases. The highest variance of maximum positional deviation is observed in the *cross* phase, as traffic participants tend to drive far into the intersection before performing the u-turn maneuver. An example of a traffic participant following this driving behavior is shown in the bottom left panel. Another traffic participant that tends to behaves differently than others is illustrated in the bottom right panel. Specifically, the truck enters the intersection using the straight-ahead lane instead of using the left-turn lane.

[17] https://sanic.dev/
[18] https://github.com/OAI/OpenAPI-Specification
[19] https://www.vvm-projekt.de/

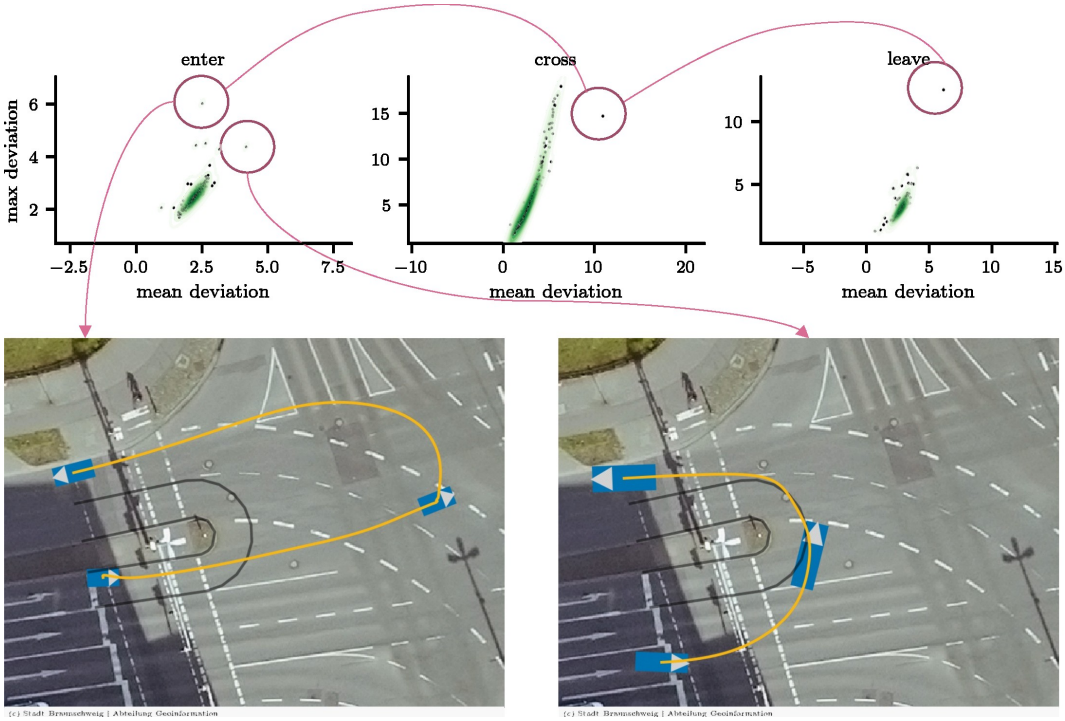


Figure 6.24: Results of the analysis of driving behavior in u-turn scenario by decomposing the maneuver into three phases (enter, cross, leave). *Top:* The distribution of the mean and maximum position deviation between the traffic participants trajectory and the lane's digital representation for the three phases. *Bottom:* The trajectories of the two traffic participants highlighted in the top panel. *Left:* The road user moves far into the intersection before performing the maneuver. *Right:* A truck enters the intersection on the straight-ahead lane to perform a u-turn maneuver. [114]

A follow-up work [142] during the VVM project aimed at extending Scenimini to continuously identify and extract crossing scenarios. In fact, the work is the foundation for the verification of crossing scenario hypotheses shown in Section 5.5.2. For that purpose, an instance of the SMP was deployed in a Tanzu Kubernetes cluster for continuous live processing with a RabbitMQ instance for inter-service communication.

The SMP services are deployed in specific groups to, for instance, target a specific use-case. The Figure 6.25 shows the data flow of the SMP core between the services via the RabbitMQ exchanges, which is extracted from its configuration file. Note that `{NS}` is a namespace placeholder for illustration purpose only. The primary purpose of the SMP core is to fetch traffic data from the database, validate the data and associate traffic participants with the routes of the intersection. In fact, the `route-backtracer` service is the implementation of the route estimation approach presented in Section 3.4. The Figure 6.25 also shows the ability to create branches, which was the motivation for using the `fanout` exchange variant by default. The results of the `route-backtracer` service are persistently stored by the `*-database-writer`, and distributed by the `*-scenario-distributor` ser-

vice according to the path the traffic participants are associated to. Note that the distribution is indicated in Figure 6.25 with the three exchanges named with `{NS}.route.path.*`, while the * is a placeholder for the route name.

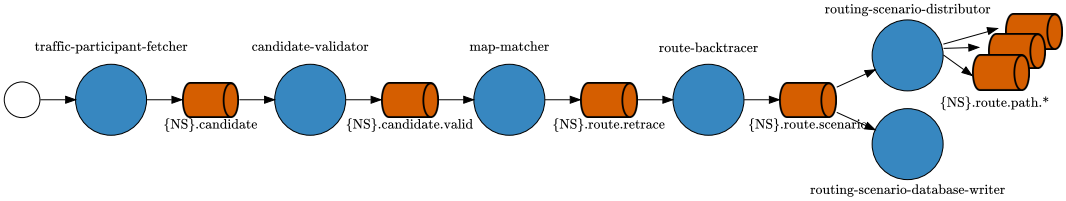


Figure 6.25: The data flow of the SMP core group between the services through the RabbitMQ exchanges visualized as acyclic directed graph.

The extension of the SMP to other use-cases based on the core group is straightforward. For the use-case of crossing scenario mining, a group of services is defined that uses the results of the core group. The aim of this group is to find traffic participants that are involved in a crossing scenario. To achieve this, traffic participants need to be matched that are associated to the two roads that define a crossing hypothesis that were identified Section 5.5.2. Figure 6.26 shows the data flow of the SMP crossing group from top to bottom. Note that `{RP}` is a placeholder for `route.path` and `{CS}` for `crossing.scenario` for illustration purpose. The crossing group consists of three service types. The `matcher` service finds matching traffic participants that are associated to the roads that define a crossing scenario hypothesis and publishes the results into an exchange dedicated to the crossing scenario. All `matcher-*` service variants illustrated in Figure 6.26 are based on the same Docker image. They only differ in how they are configured on deployment. The second service type is `pet-estimation` and it filters crossing scenario hypotheses using the SMoS PET. The results are persistently stored by the `pet-database-saver` service into the scenario database using ScenORM.

As already noted previously, docker-compose is used for configuration and deployment. In fact, the docker-compose configuration file for the SMP crossing group is created using Jinja templates[20]. This allows to automatically generate the service configuration for every matcher service. That is, the Knowledge base is queried for the potential hypotheses and for every hypothesis, the service configuration is automatically generated with the corresponding source and destination exchange as defined in Section 5.5.2. This allows to easily adapt the SMP crossing group for scenario mining on other intersections with a digital representation in OpenDRIVE.

Figure 6.27 shows the RabbitMQ exchanges that are created by the services for communication after deployment. The first column contains the names of the exchanges and the second the exchange variant. As indicated previously, the `fanout` variant is the default for all exchanges. The last two columns denote the rate of incoming and outgoing messages. The exchange naming schema follows a semantic that indicates the deployment location of the SMP, its primary purpose (live processing), the measurement site (fokr for an internal abbreviation for the AIM Research Intersection) and the name of the job. The left

[20] `https://jinja.palletsprojects.com/`

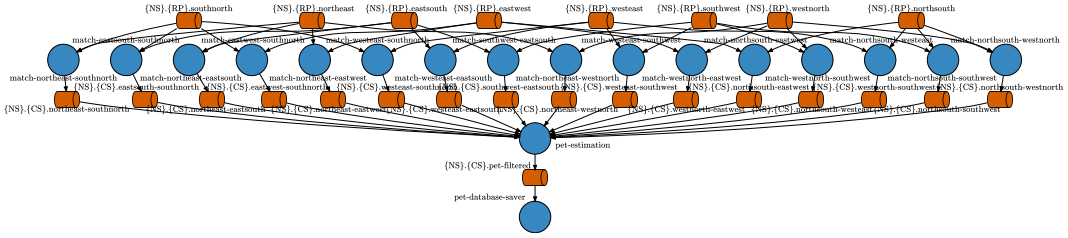


Figure 6.26: The data flow from top to bottom of the SMP crossing group between the services through the RabbitMQ exchanges visualized as acyclic directed graph.

panel of Figure 6.27 shows all exchanges of the SMP core group. Note that there is an exchange for every route as indicated in Figure 6.25. The right panel of Figure 6.27 shows the exchanges that are used by services for crossing scenario extraction. Every crossing scenario exchange contains the traffic participants that are involved in the scenario, while the traffic participants on the first route are considered as the ego participant. Since a crossing scenario is considered valid in Section 5.5.2 if the PET is lower than four seconds, the PET between traffic participants of all concrete scenarios are estimated and valid scenarios are published in the `*.pet-filtered` exchange.

tanzu.smp.live.fokr.route.path.easteast	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.eastnorth	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.eastsouth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.route.path.eastwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.route.path.northeast	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.route.path.northnorth	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.northsouth	fanout	D	0.20/s	4.8/s
tanzu.smp.live.fokr.route.path.northwest	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.southeast	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.southnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.route.path.southsouth	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.southwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.route.path.westeast	fanout	D	0.20/s	4.8/s
tanzu.smp.live.fokr.route.path.westnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.route.path.westsouth	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.path.westwest	fanout	D	0.00/s	
tanzu.smp.live.fokr.route.retrace	fanout	D	0.40/s	0.40/s
tanzu.smp.live.fokr.route.scenario	fanout	D	0.40/s	1.2/s

tanzu.smp.live.fokr.scenario.crossing.eastsouth-southnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.eastwest-southnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northeast-eastsouth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northeast-eastwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northeast-southnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northeast-westnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northsouth-eastwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northsouth-southwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northsouth-westeast	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.northsouth-westnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.pet-filtered	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.southwest-eastsouth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.westeast-eastsouth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.westeast-southnorth	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.westeast-southwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.westnorth-eastwest	fanout	D	0.00/s	0.00/s
tanzu.smp.live.fokr.scenario.crossing.westnorth-southwest	fanout	D	0.00/s	0.00/s

Figure 6.27: The RabbitMQ exchanges for inter-service communication of the SMP. *Left:* The exchanges of the SMP core focusing on the route association of traffic participants. *Right:* The exchanges for the different crossing scenario and for PET based filtering.

The extracted concrete crossing scenarios are persistently stored for later analysis using ScenORM. To achieve this, services consume from specific exchanges to save the results into an Oracle database. The concrete database tables to store the results of the SMP core group and the crossing scenario extraction group are illustrated in Figure 6.28. The table names also contain the configuration information that are part of the RabbitMQ exchange names. This enables to separate results from different instances of the SMP. It is worth noting that the SMP instance for live processing can handle restarts and recovers from temporary failures by storing the processing state in the database. An input connector forwards traffic data from the database into RabbitMQ using batch processing with fixed time intervals. For every interval, the database is queried for existing traffic participants and each published to RabbitMQ. The `*FETCHER_TRACKER_STATE` table is used to save the

processed batches and to identify the next batch to process in case of a restart of temporary failure.

Figure 6.28: A ScenORM database schema excerpt of the SMP instance for live processing.

Another work of Schicktanz *et. al.* [143] is driven by components of Scenimini that studies the effect of traffic congestions on traffic safety and efficiency. The study compares the behavior of traffic participants with and without congestion using a data set with congestion on the southern exit of the Research Intersection. To achieve this, traffic participants are associated to routes of the intersection using the virtual loop-based approach presented in Section 6.3.2 and the SMoS PET is employed as a measure of traffic safety in crossing scenarios. The study reveals that the minimum PET between traffic participants in a crossing scenario is decreased in congestions, potentially increasing risk of traffic accidents. Furthermore, a congestion on one exit of the intersection notably increases the total mean traffic delay as shown in Figure 6.29. Additionally, the study shows that congestion does not only impact the routes that are used to leave the intersection towards the direction with congestion, but also other routes of the intersection.

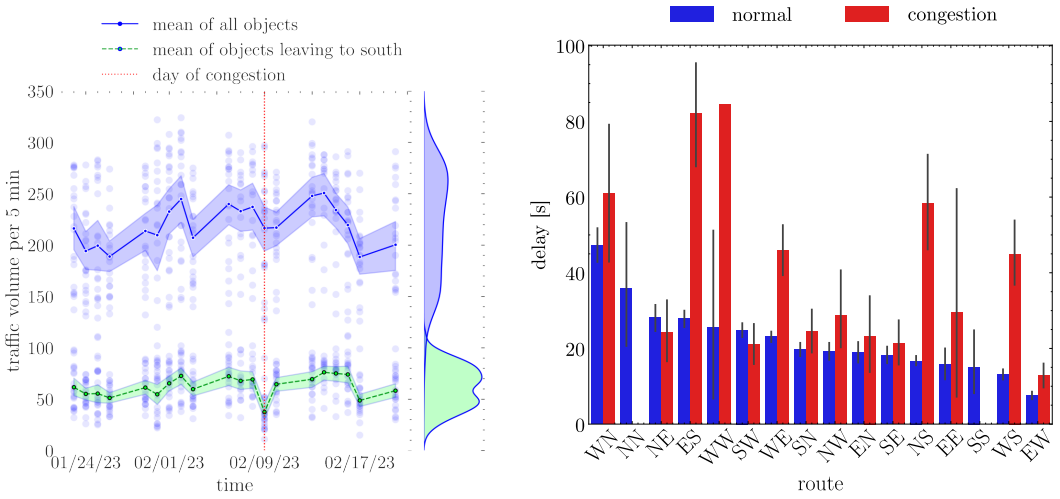


Figure 6.29: Results of the analysis of congestion impact on traffic volume and delay by Schicktanz *et. al.* [143]. *Left:* The total daily traffic volume at the intersection and the volume of routes leaving towards the exit during congestion. *Right:* The traffic delay experienced on each route with and without congestion.

6.4.2 Traffic analysis in the rural domain

The last section illustrates use-cases on the AIM Research Intersection and demonstrates the configuration and deployment of the SMP for the purpose of crossing scenario extraction. The following section will present the usage and extension of Scenimini for the VRIEDRICH project[21]. The project's aim is the observation and analysis of rural traffic to reduce the research gap in understanding rural traffic. To achieve this, two measurement campaigns were conducted, each lasting two weeks, at two rural sites. After the measurement campaigns, the data is processed using components of Scenimini.

Figure 6.30 shows two example Scenimini applications that are designed for the project. The left panel shows a web-based player for visualization of video material. Upon selection of the measurement day by the user, the application starts collecting the video material. This is indicated by the green markers below the slider. The application supports fast forward playback of the video materials at up to sixteen times faster than normal. The playback state and playback speed can be changed by keyboard keys. The slider at the bottom of the application enables to navigate to a specific point in time on the selected day. For the visualization of video material, the application communicates with the TraSceAPI of the TAP-API. Specifically, the application queries the API for the measurement campaign period. Furthermore, the application establishes a WebSocket connection to receive the video material together with details of the traffic participants that are detected by the system.

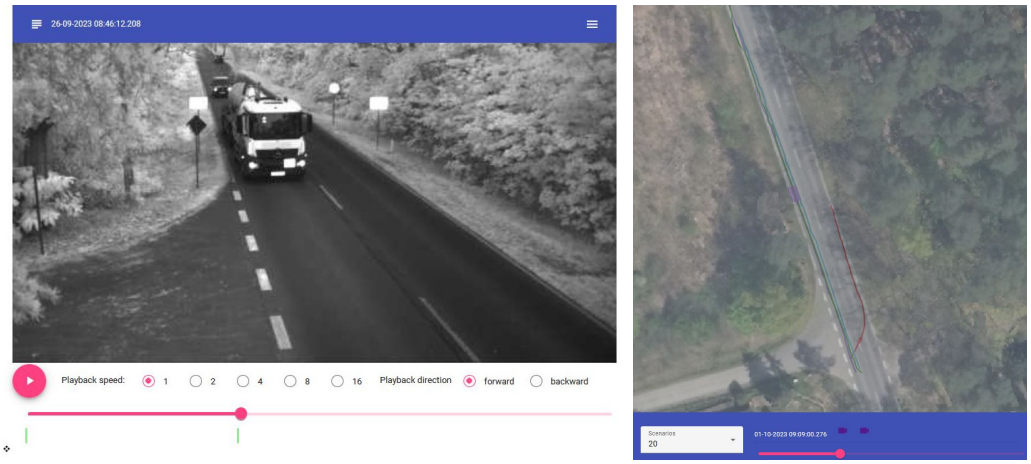


Figure 6.30: The web-based applications used in VRIEDRICH for situation and scenario analysis. *Left:* Application for video streaming of a selected camera. *Right:* Application for trajectory-based visualization of crossing scenarios. © DLR

The right panel of Figure 6.30 shows another application for scenario-based analysis. The SMP was customized for the measurement campaigns to, among others, extract crossing scenarios. The application visualizes the extracted crossing scenarios that are saved

[21] https://verkehrsforschung.dlr.de/de/projekte/vriedrich-verkehrsbeobachtung-und-analyse-im-laendlichen-raum-im-kontext-der

by the `pet-database-saver` service which is illustrated as the last service in Figure 6.26. Analogous to the first application, this application communicates with the TraSceAPI to query for the extracted crossing scenarios and for scenario-related details, such as the trajectories of the traffic participants involved in the scenario. The applications was utilized throughout the project, facilitating studies conducted based on the collected traffic data.

The study of [144] analyzed the behavior of traffic participants in crossing scenarios, among others. The SMP was adapted and deployed to extract concrete crossing scenarios. To achieve this, traffic participants are associated with the routes illustrated on a satellite image of the measurement site in Figure 6.31 using the map matching approach `MLCR` defined in (3.37), while assuming all roads are connecting roads. The routes were identified from real-world data. Since the data showed that the most potential crossing interactions occur on the west-north and north-west routes, those were extracted using the SMP. For the realization of the scenario mining process, only the service implementing the `MLCR` task needed to be implemented, while the other services of the SMP core and crossing group could be reused. The middle panel of Figure 6.31 illustrates the distribution of crossing points of the traffic participants involved in a scenario. The number of scenarios is color-encoded from black to blue. Based on the extracted crossing scenarios, the study found out, among others, that traffic participants on the prioritized road often actively prevent critical situation by decelerating if a traffic participant from west merges. Two examples are shown in the right panel of Figure 6.31 where the merging traffic participant ignores the right-of-way of the traffic participant on the federal road. The latter needs to break to prevent a collision as indicated by the rear brake lights.

Figure 6.31: *Left:* The routes of the measurement site. *Middle:* The distribution of crossing points on the measurement site for the north-west scenario. `Right:` Examples of breaking traffic participants in crossing scenarios on the federal road. [144]

In another study [144], Scenimini was extended to extract overtaking scenarios in a village entrance with rural character (see left panel of Figure 6.32). In that study, the primitive-based approach discussed in Chapter 3 was employed to relate traffic participants on the federal road to each other using the longitudinal and lateral relations defined in Table 3.1. The description of traffic in terms of the primitives allowed to identify and extract overtaking scenarios. The aim of that study is the analysis of the driving behavior in overtaking scenarios, particularly in terms of the difference velocity during the maneuvers

and the starting positions (see Figure 6.32) focusing on the interaction between motorized vehicles and bicycles, as it is a complex, critical but often occurring maneuver [145], [146]. The study indicates that, in most cases, vehicle overtaking cyclists comply with the minimum safety distance. Furthermore, overtaking vehicles tend to have significantly higher speeds than the overtaken vehicles. Most of these vehicles initiate the maneuver within the town while heading out of town and exceed the maximum speed limit allowed in the village.

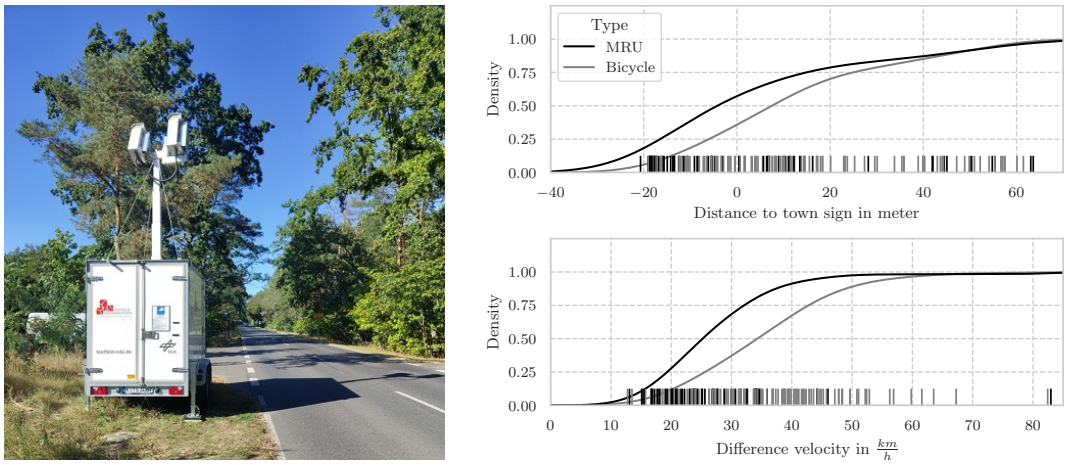


Figure 6.32: Traffic data acquisition at a village entrance. *Left:* The mobile measurement station. *Right:* The cumulative distribution functions of the overtaking maneuver start position relative to the town sign and the difference velocity during the overtaking maneuvers. [144]

6.4.3 Test Bed Lower Saxony

The DLR Test Bed Lower Saxony makes it possible to analyze the driving behavior of traffic participants on a German highway. For this purpose, 71 masts, each equipped with two stereo camera systems, are installed along 7.5 kilometers of the A39 between Cremlingen and the Wolfsburg/Königslutter junction.

The SMP of Scenimini is extended and utilized in [147] to extract a variant of space-sharing conflict scenarios that occur on highways: the on-ramp scenario. This scenario is especially of interest and discussed in the academic domain for several years, since traffic participants involved in that scenario need to cooperate the resolve the situation. Moreover, the road user on the acceleration lane needs to perform a lane change maneuver to merge into the main traffic, which is a risky maneuver and a typical source of traffic delays [148].

The study [147] examines the driving behavior of the traffic participants as they merge onto the main lane, including their use of the acceleration lane, the timing of their lane-change maneuver, and the accepted gap times (refer to Section 2.4.2) for merging using traffic data collected near Cremlingen (see Figure 6.33). The study shows that the

primitive-based approach for lane-change identification and extraction proposed in Chapter 3 can be utilized to robustly extract on-ramp scenarios without reconfiguring the HMM that is used to represent the trajectory in the domain of primitives.

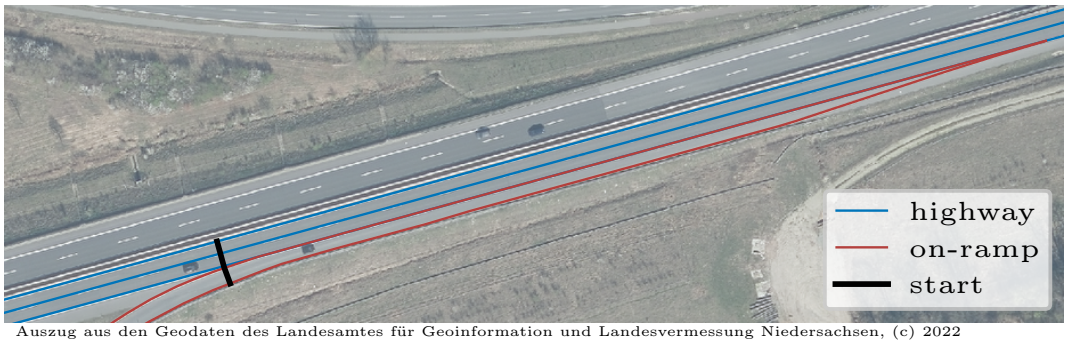


Figure 6.33: The on-ramp section of the Test Bed Lower Saxony near Cremlingen with the digital road representation. [114]

Figure 6.34 shows a Scenimini web-based application that displays the extracted scenarios for the on-ramp section near Cremlingen and a second on-ramp region near Scheppau. Every row of the table shows basic details of a scenario, such as the start and end time and the on-ramp region. The Scenimini application uses the TraSceAPI of the presented TAP-API in Section 6.3 to query the scenarios from the Scenario database that are extracted using the SMP. This tools is used to access the video data of the cameras located in the specific region and thus for video-based scenario analysis.

TFNDS On-Ramp Scenario Overview

This page contains all extracted on-ramp scenarios on the TFNDS and provides information about the scenario and links to the scenario videos. Happy exploring!

Show 10 entries Search:

id_ego_participant	id_scenario	name	t_end	t_start	region	link
1	1	cremlingen_on_ramp_scenario	2021-09-11 16:00:21.674657	2021-09-11 16:00:20.234657	Cremlingen	
3	2	cremlingen_on_ramp_scenario	2021-09-11 16:03:42.434657	2021-09-11 16:03:20.234657	Cremlingen	
12	3	cremlingen_on_ramp_scenario	2021-09-11 16:12:27.874657	2021-09-11 16:12:00.714657	Cremlingen	
13	5	cremlingen_on_ramp_scenario	2021-09-11 16:12:26.394657	2021-09-11 16:11:59.514657	Cremlingen	
17	4	cremlingen_on_ramp_scenario	2021-09-11 16:12:56.554657	2021-09-11 16:12:34.514657	Cremlingen	
21	21	scheppau_on_ramp_scenario	2021-09-11 16:02:50.514657	2021-09-11 16:02:44.394657	Scheppau	
22	35	scheppau_on_ramp_scenario	2021-09-11 16:02:46.114657	2021-09-11 16:02:44.714657	Scheppau	
24	22	cremlingen_on_ramp_scenario	2021-09-11 16:06:39.114657	2021-09-11 16:06:16.274657	Cremlingen	
26	23	scheppau_on_ramp_scenario	2021-09-11 16:05:55.474657	2021-09-11 16:05:36.754657	Scheppau	
28	24	scheppau_on_ramp_scenario	2021-09-11 16:03:44.994657	2021-09-11 16:03:39.714657	Scheppau	

Showing 1 to 10 of 27,491 entries Previous 1 2 3 4 5 ... 2750 Next

Figure 6.34: A web-based interface showing the extracted merging scenario on the Test Bed Lower Saxony for different regions. © DLR

An example page for an on-ramp scenario is illustrated in Figure 6.35 with scenario details shown at the top of the page, while the remainder of the page shows the videos of the cameras that are located in the scenario region. Every video is streamed by the TraSceAPI

using the Motion JPEG codec since this format is natively supported by many modern web-browser.

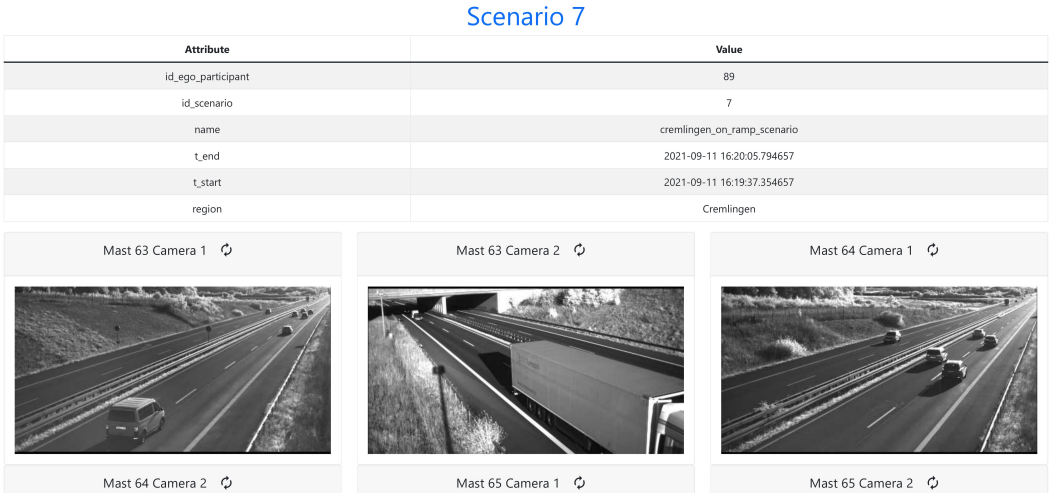


Figure 6.35: Details of an extracted merging scenario on the Test Bed Lower Saxony including videos of the cameras located in scenario region. © DLR

Scenimini enables the analysis of scenarios in Python by providing scenario information through the TAP-API, particularly by ScenORM. This was utilized in the study mentioned above [147] to analyze driving behavior in on-ramp scenarios, despite the use of video-based analysis. The work follows the approach by Qi *et. al.* [149] and specializes the on-ramp scenario into four variants by taking into the account the position of the two closest traffic participants on the adjacent main lane. This is based on the assumption that the merge behavior of the vehicle in the acceleration lane would be influenced by the occupancy level of the neighboring lane. Additionally, the study aims to investigate the time gaps between traffic participant in the three different on-ramp scenarios. Figure 6.36 illustrates the four on-ramp scenario specializations *free*, *in front*, *behind* and *into* by considering the longitudinal location of the ego participant relative to the challengers on the main lane. In fact, the study follows the primitive-based approach for the representation of traffic data using primitives and the identification of maneuvers presented in Chapter 3. But, instead of maneuver classification, the approach is utilized for scenario categorization. To achieve this, the trajectories of all traffic participants are represented in the Frenét coordinate frame (see Section 2.1.2) defined by the lane marking, and the scene at the moment the ego participant crosses the left lane marking of the acceleration lane is represented in the primitive domain. This domain is defined by the longitudinal positional relationship from Table 3.1 between the ego participant and the traffic participants on the adjacent lane as illustrated in the overtaking scenario example from Figure 3.3. Assuming that traffic participants do not collide, the primitive `Next to` can be neglected and only the other two primitives `Behind` and `In front` are employed to represent the scene in the primitive domain. The work uses the PET to define the context, which allows to represent

the scene in the primitive domain (see Section 3.2). For the categorization of the scenario, a rule-based approach is employed based on the representation in the primitive domain.

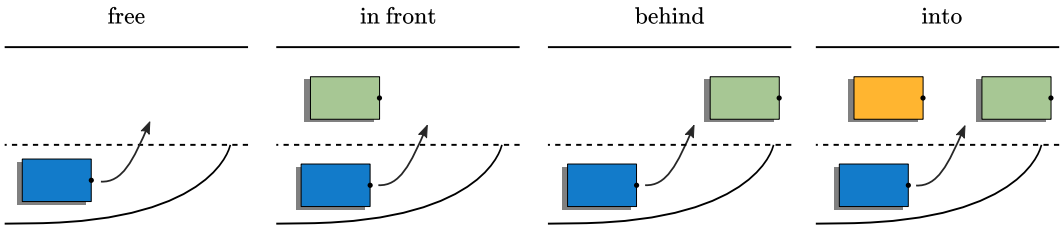


Figure 6.36: On-ramp merging scenarios are specialized into four classes according to the position of challengers (based on [114]).

That is, let $s \in \mathcal{S}$ be an instance of an on-ramp scenario and $\mathcal{S} = \{\mathcal{S}^{\emptyset}_{\emptyset}, \mathcal{S}^{+}, \mathcal{S}_{-}, \mathcal{S}^{+}_{-}\}$ be the set of categorized on ramp scenarios as illustrated in Figure 6.36 with $\mathcal{S}^{\emptyset}_{\emptyset}$ the *free*, $\mathcal{S}^{+}$ *in front*, $\mathcal{S}_{-}$ behind and $\mathcal{S}^{+}_{-}$ the *into* scenario. Additionally, the primitive domain $\mathbb{D}$ is defined by the longitudinal positional relationship $\mathbb{D} = \{\text{Behind, In front}\}$ and let $C = c_1, c_2, \ldots$ be the challengers on the main lane with $c \in C$ as the challenger's trajectory. The challenger trajectory contains the evolution of the front right and rear right position, while the ego trajectory the front left and rear left position. Then, $\mathcal{P}_c = \{\mathbf{p}_1, \mathbf{p}_2, \mathbf{p}_3, \mathbf{p}_4\}$ are the four points where the trajectory of the challenger c and the ego participant intersect. For each of the intersection points $p \in \mathcal{P}$ let $t_{ego}(p)$ and $t_c(p)$ be the arrival time of the ego and challenger participant such that the difference in arrival time $\Delta t(p)$ at p or the PET is estimated according to (2.8) as

$$\Delta t(p) = t_{ego}(p) - t_c(p) \tag{6.1}$$

so that $\Delta T_c = \{\Delta t_1, \Delta t_2, \Delta t_3, \Delta t_4\}$ are the differences in time of arrival for the four intersection points. The definition of the PET from Section 2.4.3 is thus extended for this use case with multiple intersection points of the candidate $c \in C$ and the ego participant as

$$\text{PET}(c) = \underset{\Delta t \in T_c}{\text{argmin}} |\Delta t| \tag{6.2}$$

with $|\Delta t|$ the absolute time difference. The context $K = \{\text{PET}(c_1), \text{PET}(c_2), \ldots, \text{PET}\left(c_{|C|}\right)\}$ that is used to represent the scene in the primitive domain is thus defined as the PET to all candidates. Furthermore, the transformation function $t(k)$ that represents a scene in the primitive domain with $k \in K$ is defined as

$$y = t(k) = \begin{cases} \texttt{Behind} & \text{if } k > 0 \\ \texttt{In front} & \text{if } k < 0 \end{cases} \tag{6.3}$$

assuming no collision of traffic participants with y as a component of the scene in the primitive domain $\mathbf{y} = \left[y_1, y_2, \ldots, y_{|C|}\right]^T$. Given the set of on ramp scenario variants $\mathcal{S}$ and the representation of the moment in time of the ego vehicle crossing the left lane marking

of the acceleration lane in the primitive domain $\mathbf{y}$, $f : \mathbf{y} \rightarrow \mathcal{S}$ represents the rule-based function to categorize a scenario. It is defined as

$$f(Y) = \begin{cases} s \in \mathcal{S}_{\emptyset}^{\emptyset} & \text{if} \quad |\mathbf{y}| = \emptyset \\ s \in \mathcal{S}^{+} & \text{if} \quad \forall y_i \in \mathbf{y} : y_i = \texttt{Behind} \\ s \in \mathcal{S}_{-} & \text{if} \quad \forall y_i \in \mathbf{y} : y_i = \texttt{In front} \\ s \in \mathcal{S}_{-}^{+} & \text{if} \quad \forall d \in \mathbb{D} : \exists d \in y_i \end{cases} \tag{6.4}$$

such that a scenario s is categorized as `free` if there are no challengers, as `in front` if all challengers are `Behind`, `behind` if all challengers are `In front` and as `into` if there are challengers that are `In front` and `Behind` of the ego vehicle.

The approach for scenario categorization is used in [114] to analyze the merging behavior of traffic participants. The study uses $N = 144$ on-ramp scenarios, and considered traffic participants as challengers if the PET was lower than 10 seconds. Moreover, the projection into the road reference line coordinate system (see Section 2.1.2) defined by the lane marking is used to estimate the distance of each traffic participant to the start marker shown in Figure 6.33.

Figure 6.37 shows the distribution of the absolute PET value for the different scenario categories. The sum of the absolute PET value to the two closest traffic participants is illustrated for the `into` category, which is related to the SMoS accepted gap time. The results indicate that in the `in front` scenario, traffic participants tend to merge into the main lane with a smaller gap time compared to the `behind` scenario. Additionally, the accepted gap time for the `into` scenario is uniformly distributed within [3, 15.1] seconds. The study also found that traffic participants do not use the full extent of the acceleration lane. Most of the merging maneuvers analyzed in the study (79.56%) start in the first half of the lane, and 39.42% have already merged onto main lane in the first half of the acceleration lane.

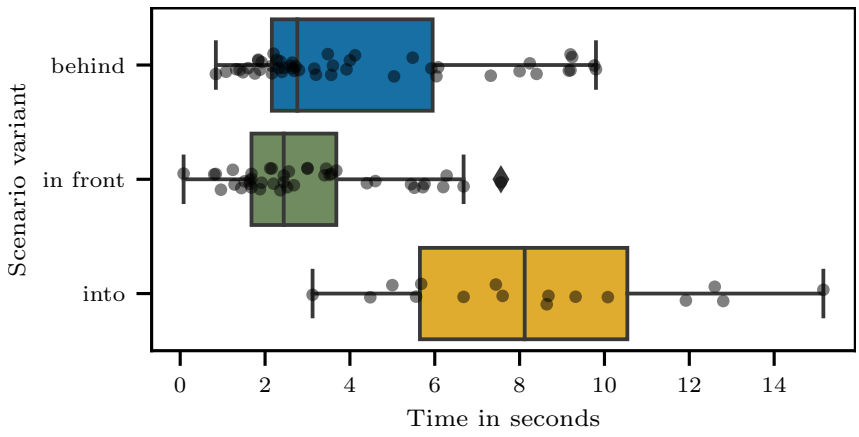


Figure 6.37: Distribution of the PET for all identified on-ramp scenarios of type `behind` and `in front` and the accepted gap time for `into`. [147]

6.5 Summary

This chapter presented as platform that connects the methods and methodologies presented throughout the previous chapters for scenario-based analysis of real-world traffic data.

At first, the architecture of the platform is outlined, emphasizing the components of the three layers and their interconnection. Then, the Scenario mining pipeline (SMP) is presented, which processes traffic data for scenario extraction. The scenario mining process, that is followed in this work, is used to organize tasks and to guide in the configuration and deployment of the SMP. Additionally, the SMP development lifecycle is presented, that illustrates the scenario mining process as an iterative approach, while also highlighting considerations for effective implementation of the tasks along the scenario mining process as SMP services.

Furthermore, the chapter presents APIs that facilitate the access of platform components to the data layer. The Scenario object-relational mapping (ScenORM) library is introduced that provides a object- and scenario-oriented view on traffic data. In addition, the Traffic situation analysis and interpretation (TASI) is presented, which is a Python-based library for traffic data representation and performant traffic data analysis.

The chapter also illustrates the versatility of the platform in a variety of practical applications, demonstrating its adaptability to diverse applications domains and operational contexts. The platform was employed in the VRIEDRICH project for the analysis of rural traffic, in the VVM research project for the extraction and analysis of crossing scenarios on the AIM Research Intersection, and for the analysis of on-ramp merging scenarios on the Testbed Lower Saxony. The platform has become an important pillar of the DLR Institute of Transportation Systems' Department of Information Acquisition and Model Design. It serves as the foundation for subsequent research projects and studies.

Chapter 7

Conclusion

In the following, the thesis is briefly summarized in Section 7.1. Thereafter, the results of this work will be discussed in relation to the research question and objectives in Section 7.2. Finally, an outlook on potential follow-up questions will be provided in Section 7.3.

7.1 Summary

For the safety argumentation of automated driving functions traditional methods are limited in their applicability. This is primarily due to the complexity of the systems and the environments in which they operate. The scenario-based testing approach aims to address this, by breaking down this overall complexity into smaller manageable sub-problems. This enables to assess the driving functions in these relevant scenarios to argue for the systems safety. Scenarios derived from real-world data are particularly valuable for this purpose. However, a challenge lies in extracting these scenarios from traffic data collected, for example, from vehicle fleets or stationary infrastructure, as these continuously gather data that needs to be processed.

This thesis introduced a framework that enables the systematic extraction of scenarios from real-world traffic data. This framework is modular and comprises various components for the continuous identification, extraction, and analysis of scenarios. A hierarchical four-layer data model of scenes, primitives, maneuvers and scenario is utilized, and the thesis presented methods to describe entities at different levels of this model and identify them in real-world traffic data. For this purpose, Chapter 3 presented an approach to translate traffic data into a semantic level. This involves converting sensor data or environmental information that is typically described numerically into terms and concepts from human language, which are termed primitives in this work. While primitives were already used for semantic analysis of driving behavior [150][44], this work showed that more complex concepts such as maneuvers can be robustly derived from it. The proposed methodology was validated through two case studies with different use-cases, indicating its adaptability to various real-world problems, as also demonstrated by different application domains of the Scenimini platform in Section 6.4.

In Chapter 4, an alternative approach for maneuver identification is presented. This method is based on a digital map of the road network in OpenDRIVE. The underlying assumption is that intersections in road networks are designed to allow traffic participants to adjust their travel directions, thus facilitating certain infrastructural maneuvers. In

contrast to trajectory data of traffic participants, this method annotates the roads of the digital map with corresponding maneuvers. The association of traffic participants with the roads of the road network enables the identification of the respective maneuver. In Chapter 5, a methodology for the systematic identification of specific scenarios in urban areas is presented. In particular, it outlines a method for detecting space-sharing conflict scenarios. The approach employs ontologies in order to automatically generate scenario hypotheses based on expert knowledge and a digital representation of an intersection. The efficacy of this approach is assessed using a publicly available digital map of Braunschweig, Germany, produced by the AIM DLR Research Intersection. The digital map is used to derive hypotheses, which are then validated using real-world trajectory data. The findings indicate that not all hypotheses are valid. Moreover, the results suggest that the relative frequency of scenarios can be used as an indicator to identify cases where road users exhibit atypical behavior. For example, red light running scenarios were identified that can lead to potentially critical encounters.

For the continuous identification, extraction and analysis of scenarios from real-world traffic data, the methods proposed thus far are integrated into a modular three tier platform. The Scenario mining platform (Scenimini) is presented in Chapter 6. This chapter introduces the scenario mining process, among other things. This process outlines the steps to represent real-world traffic data as scenarios and serves as the foundation for organizing SMPs. These pipelines are modular distributed applications consisting of services, each encapsulating tasks along the process. Unlike the scenario mining process, the development and deployment of these SMP for scenario identification, extraction, and analysis follows a cyclical process. These pipelines can be designed for various problem scenarios, taking advantage of a service-oriented architecture that allows reuse of existing services.Furthermore, several interfaces are introduced for the management and processing of traffic data within the services, as well as for database-based scenario management. This includes the Python library TASI, which provides models for traffic analysis based on established libraries. These data models allow the homogenization of traffic data from different sources such as floating vehicles or stationary infrastructure. Additionally, TASI serves as a toolbox that allows, for instance, the conversion of concrete scenarios into other formats such as OpenSCENARIO for simulation-based analysis. The versatility and adaptability of Scenimini has been demonstrated through various applications addressing different problems or questions, such as the identification of concrete crossing scenarios that are used in Chapter 5, in different environments.

7.2 Discussion

This thesis addresses the problem of representing real-world traffic data continuously acquired by infrastructure as scenarios. The main contributions are a framework for the semantic representation of traffic data, a systematic approach for scenario identification and a platform for continuous scenario identification, extraction and analysis.

Scenarios are typically initially defined using terms from human language. To identify these scenarios in real-world data, they need to be translated into these terms. Objec-

tive O.III outlines a methodology that addresses this issue and in Section 3 a solution is provided. Specifically, a methodology is proposed that can serve as a guideline to identify the components that need to be defined in order to represent traffic data in terms of primitives.

In the two case studies, this methodology was applied exemplarily, demonstrating that traffic data can be represented in terms of primitives using a variety of methods. The objective was thus successfully achieved. Furthermore, the case studies demonstrate different approaches to defining maneuvers based on primitives. For instance, in Section 3.3, the lane change maneuver is described using primitives, and a HMM is employed to map traffic data onto these primitives. For identifying these maneuvers, they were defined within the domain of primitives using maneuver signatures. In Section 3.4, the methodology is employed to calculate the route of traffic participants in urban areas. The roads of the network serve as the domain of primitives, and the possible maneuvers correspond to the routes that traffic participants can take to cross intersections. Unlike the first case study, the maneuvers are not predefined, but are derived from the digital representation of the road network. In this study, a dynamically extending directed acyclic graph is employed to represent the primitives and the relationships among them. The results of both case studies demonstrate that this methodology can describe different phenomena in traffic data, thereby achieving objective O.IV. The continuous and long-term processing of real-world traffic data is essential for the analysis of traffic events, especially of those that are rare, and the collection of concrete real-world scenarios. Therefore, an appropriate technical framework is required. This framework must be flexible in order to adapt to various problems and research questions that may focus on different scenarios, and capable of processing continuous traffic data streams. This challenge is articulated in RQ.I. Chapter 5 introduces an approach to systematically identify scenarios, thereby addressing objective O.I. The method was demonstrated using real-world traffic data from the DLR AIM Research Intersection for the purpose of deriving space-sharing conflict scenarios. The efficacy of the presented approach was demonstrated through the analysis of empirical trajectory data and the identification of potential crossing scenarios. Therefore, objective O.I can be considered achieved. Furthermore, Chapter 6 introduces a platform that enables the continuous representation of traffic data in terms of scenarios, thereby addressing objective O.II. The flexibility and scalability of the platform are demonstrated by its application to address various problems in different environments using traffic data collected by quasi-stationary and stationary infrastructure. This objective can also be considered achieved.

7.3 Outlook

The presented methods and technical frameworks enable addressing various related research questions. However, both the methods and particularly the Scenario mining platform (Scenimini) as a whole can be expanded in future work. The following section provides a brief overview of potentially relevant future work.

The primitive-based approach for traffic data representation enables the description of states, relations and phenomena on a semantical level. In future works, this approach could be extended by representing primitives in ontologies. This approach would allow the formal description of more complex phenomena in which several phenomena are related to each other in time [151]. Additionally, building upon existing concepts and relationships defined in this work, future research could leverage concepts from other ontologies, such as the A.U.T.O Ontology [152], which already defines safety-related concepts. Integrating these concepts could enrich the framework with established domain knowledge and facilitate broader interoperability and consistency in scenario representation and analysis.

The presented platform for scenario mining enables continuous processing of traffic data. This is particularly valuable for identifying rare phenomena, which are situated at the top of the safety pyramid (see Figure 2.7). This was exemplified in Chapter 5, where traffic participants committing red light violations were identified. This capability is crucial for the development of automated vehicles, as they must handle human errors in the future. Furthermore, in Section 6.3.2, it was shown that concrete scenarios can be converted into the OpenSCENARIO format. The continuous processing of traffic data and thus the creation of a comprehensive database of scenarios enable the description of logical scenarios. This was demonstrated, for example, in Section 6.4.3 for the scenario of merging onto the highway, where parameter distributions based on extracted concrete scenarios were determined. These could be used to describe the logical scenario. These are valuable for the simulation-based development and testing of automated driving functions. Logical scenarios can be used to derive concrete scenarios by sampling from the parameter distributions. A major advantage of the simulation-based approach is the thorough testing of systems in rare and critical scenarios identified in real traffic data.

Another potential topic for future works is the data-driven definition of Operational Domains (ODs) using traffic data captured by stationary infrastructure. Typically, ODs for automated driving functions are defined using expert knowledge. AVs will need to assess whether they remain within the specified Operational Domain as described by the current Operational Domain. Testbeds could assist in this process by defining and communicating the current Operational Domain to AVs. Continuous collection of traffic data through stationary infrastructure across different times of day and seasons, and mapping this data at a semantic level, could potentially identify various ODs within the respective measurement area. However, further work is required to determine whether primitives can be mapped onto terms used to describe the Operational Domain (OD).

For the extraction of scenarios from real-world traffic data, the Scenario mining pipelines (SMPs) need to be configured and deployed. For this purpose, docker-compose files are currently utilized and converted for the deployment in a Kubernetes cluster. However, this docker-compose based approach is considered complex, especially for developers that are unfamiliar with this technology. This complexity is even more pronounced when configuring deployment on the Kubernetes cluster. Moreover, the platform lacks an integrated solution for monitoring the various SMPs. The design of another user interface for Scenimini that enables the management and monitoring of Scenimini in a central place may be subject of a follow-up work. This interface should also facilitate lifecycle management of the various SMPs, including the configuration, deployment and monitoring of the in-

stances. In follow-up works, it may be worth investigating if a low-code or even no-code solution, technology that emerged could be developed to solve this problem.

It is worth noting that the core of TASI was released into the public domain during the preparation of this work, hoping to create value for the academic and industrial domains. The core version includes the presented data structures with some extensions regarding visualization and data exchange along with data from the DLR AIM Research Intersection. It is planned to expand the scope of the library in the future to incorporate additional functionality. This includes the estimation of SMoS and the generation of scenarios from real-world traffic data, as well as the addition of interfaces to support other publicly available datasets and data formats.

References

[1] F. M. Bartz, "Mobilitätsbedürfnisse und ihre satisfaktoren. die analyse von mobilitätstypen im rahmen eines internationalen segmentierungsmodells," Ph.D. dissertation, Universität zu Köln, Köln, 2015.

[2] A. Alessandrini, A. Campagna, P. D. Site, F. Filippi, and L. Persia, "Automated vehicles and the rethinking of mobility and cities," *Transportation Research Procedia*, vol. 5, pp. 145–160, 2015, ISSN: 23521465. DOI: 10.1016/j.trpro.2015.01.002.

[3] K. Wehnemann and K. Schultz, *Treibhausgas-projektionen 2024: Ergebnisse kompakt*, Umweltbundesamt, Ed., Dessau-Roßlau, 2024.

[4] E. Szczechowicz, T. Dederichs, and A. Schnettler, "Regional assessment of local emissions of electric vehicles using traffic simulations for a use case in germany," *The International Journal of Life Cycle Assessment*, vol. 17, no. 9, pp. 1131–1141, 2012, ISSN: 0948-3349. DOI: 10.1007/s11367-012-0425-8.

[5] A. D. Beza, M. Maghrour Zefreh, and A. Torok, "Impacts of different types of automated vehicles on traffic flow characteristics and emissions: A microscopic traffic simulation of different freeway segments," *Energies*, vol. 15, no. 18, p. 6669, 2022. DOI: 10.3390/en15186669.

[6] B. Zhao, Y. Lin, H. Hao, and Z. Yao, "Fuel consumption and traffic emissions evaluation of mixed traffic flow with connected automated vehicles at multiple traffic scenarios," *Journal of Advanced Transportation*, vol. 2022, pp. 1–14, 2022, ISSN: 0197-6729. DOI: 10.1155/2022/6345404.

[7] E. Aria, J. Olstam, and C. Schwietering, "Investigation of automated vehicle effects on driver's behavior and traffic performance," *Transportation Research Procedia*, vol. 15, pp. 761–770, 2016, ISSN: 23521465. DOI: 10.1016/j.trpro.2016.06.063.

[8] T. M. Gasser, "Fundamental and special legal questions for autonomous vehicles," in *Autonomous driving*, M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner, Eds., Berlin and Heidelberg: Springer Open, 2016, pp. 523–551, ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8_25.

[9] P. Lin, "Why ethics matters for autonomous cars," in *Autonomous driving*, M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner, Eds., Berlin and Heidelberg: Springer Open, 2016, pp. 69–85, ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8_4.

[10] M. Gyllenhammar, R. Johansson, F. Warg, D. Chen, H.-M. Heyn, M. Sanfridson, J. Söderberg, A. Thorsén, and S. Ursing, "Towards an operational design domain that supports the safety argumentation of an automated driving system," in *10th European Congress on Embedded Real Time Systems (ERTS 2020)*, 2020.

[11] P. Weissensteiner, G. Stettinger, S. Khastgir, and D. Watzenig, "Operational design domain-driven coverage for the safety argumentation of automated vehicles," *IEEE Access*, vol. 11, pp. 12 263–12 284, 2023, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3242127.

[12] C. Hydén and L. Linderholm, "The swedish traffic-conflicts technique," in *International Calibration Study of Traffic Conflict Techniques*, E. Asmussen, Ed., Berlin, Heidelberg: Springer Berlin Heidelberg, 1984, pp. 133–139, ISBN: 978-3-642-82109-7.

[13] W. Wachenfeld and H. Winner, "The new role of road testing for the safety validation of automated vehicles," in *Automated Driving*, D. Watzenig and M. Horn, Eds., Cham: Springer International Publishing, 2017, pp. 419–435, ISBN: 978-3-319-31893-6. DOI: 10.1007/978-3-319-31895-0_17.

[14] A. Laureshyn and A. Varhelyi, *The Swedish Traffic Conflict Technique: observer's manual*. 2018.

[15] E. de Gelder and O. Op den Camp, "How certain are we that our automated driving system is safe?" *Traffic Injury Prevention*, vol. 24, no. sup1, S131–S140, 2023.

[16] H. Winner, K. Lemmer, T. Form, and J. Mazzega, "Pegasus—first steps for the safe introduction of automated driving," in *Road Vehicle Automation 5*, ser. Lecture Notes in Mobility Ser, G. Meyer, Ed., Cham: Springer, 2019, pp. 185–195, ISBN: 978-3-319-94895-9. DOI: 10.1007/978-3-319-94896-6_16.

[17] R. Galbas, J. Reich, H. Schittenhelm, and N. Wagener, "Safeguarding methods for complex traffic scenarios for approval of automated driving functions," *ATZ worldwide*, vol. 124, no. 9, pp. 56–61, 2022. DOI: 10.1007/s38311-022-0857-0.

[18] A. Etemad, "Adaptive: Automated driving applications and technologies for intelligent vehicles," in *Automated Driving*, D. Watzenig and M. Horn, Eds., Cham: Springer International Publishing, 2017, pp. 535–540, ISBN: 978-3-319-31893-6. DOI: 10.1007/978-3-319-31895-0_23.

[19] H. Elrofai, J.-P. Paardekooper, E. de Gelder, S. Kalisvaart, and O. Op den Camp, *Scenario-based safety validation of connected and automated driving*, Netherlands Organization for Applied Scientific Research, Ed., 2018.

[20] T. Winkle, "Development and approval of automated vehicles: Considerations of technical, legal, and economic risks," in *Autonomous driving*, M. Maurer, J. C. Gerdes, B. Lenz, and H. Winner, Eds., Berlin and Heidelberg: Springer Open, 2016, pp. 589–618, ISBN: 978-3-662-48845-4. DOI: 10.1007/978-3-662-48847-8_28.

[21] C. Neurohr, L. Westhofen, T. Henning, T. de Graaff, E. Mohlmann, and E. Bode, "Fundamental considerations around scenario-based testing for automated driving," in *2020 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2020, pp. 121–127, ISBN: 978-1-7281-6673-5. DOI: 10.1109/IV47402.2020.9304823.

[22] W. Huang, K. Wang, Y. Lv, and F. Zhu, "Autonomous vehicles testing methods review," in *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, Piscataway, NJ: IEEE, 2016, pp. 163–168, ISBN: 978-1-5090-1889-5. DOI: 10.1109/ITSC.2016.7795548.

[23] F. Khan, M. Falco, H. Anwar, and D. Pfahl, "Safety testing of automated driving systems: A literature review," *IEEE Access*, vol. 11, pp. 120 049–120 072, 2023, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2023.3327918.

[24] A. Pütz, A. Zlocki, J. Bock, and L. Eckstein, "System validation of highly automated vehicles with a database of relevant traffic scenarios," in *12th ITS European Congress*, ITS European Congress, 2017.

[25] D. Zhao, Y. Guo, and Y. J. Jia, "Trafficnet: An open naturalistic driving scenario library," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–8, ISBN: 978-1-5386-1526-3. DOI: 10.1109/ITSC.2017.8317860.

[26] C. Xu, W. Ding, W. Lyu, Z. LIU, S. Wang, Y. He, H. Hu, D. Zhao, and B. Li, "Safebench: A benchmarking platform for safety evaluation of autonomous vehicles," in *Advances in Neural Information Processing Systems*, S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, Eds., vol. 35, Curran Associates, Inc, 2022, pp. 25 667–25 682.

[27] Q. Li, Z. Peng, L. Feng, Z. Liu, C. Duan, W. Mo, and B. Zhou, "Scenarionet: Open-source platform for large-scale traffic scenario simulation and modeling," in *Advances in Neural Information Processing Systems*, A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, Eds., vol. 36, Curran Associates, Inc, 2023, pp. 3894–3920.

[28] R. Gruner, P. Henzler, G. Hinz, C. Eckstein, and A. Knoll, "Spatiotemporal representation of driving scenarios and classification using neural networks," in *28th IEEE Intelligent Vehicles Symposium*, Piscataway, NJ: IEEE, 2017, pp. 1782–1788, ISBN: 978-1-5090-4804-5. DOI: 10.1109/IVS.2017.7995965.

[29] L. Ries, J. Langner, S. Otten, J. Bach, and E. Sax, "A driving scenario representation for scalable real-data analytics with neural networks," in *IV19*, Piscataway, New Jersey: IEEE, 2019, pp. 2215–2222, ISBN: 978-1-7281-0560-4. DOI: 10.1109/IVS.2019.8813881.

[30] P. Sun, H. Kretzschmar, X. Dotiwalla, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," IEEE, 2020. DOI: 10.1109/CVPR42600.2020.00252.

[31] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari, *Nuplan: A closed-loop ml-based planning benchmark for autonomous vehicles*, 2021.

[32] T. Yin, X. Zhou, and P. Krahenbuhl, "Center-based 3d object detection and tracking," in *2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, Piscataway, NJ: IEEE, 2021, pp. 11 779–11 788, ISBN: 978-1-6654-4509-2. DOI: 10.1109/CVPR46437.2021.01161.

[33] L. Kong, X. Xu, J. Ren, W. Zhang, L. Pan, K. Chen, W. T. Ooi, and Z. Liu, *Multi-modal data-efficient 3d scene understanding for autonomous driving*, 2024.

[34] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in *Computer vision - ECCV 2016*, ser. Lecture Notes in Computer Science, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., vol. 9912, Cham: Springer, 2016, pp. 549–565, ISBN: 978-3-319-46483-1. DOI: 10.1007/978-3-319-46484-8_33.

[35] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highd dataset: A drone dataset of naturalistic vehicle trajectories on german highways for validation of highly automated driving systems," in *2018 IEEE Intelligent Transportation Systems Conference*, Piscataway, NJ: IEEE, 2018, pp. 2118–2125, ISBN: 978-1-7281-0321-1. DOI: 10.1109/ITSC.2018.8569552.

[36] R. Krajewski, T. Moers, J. Bock, L. Vater, and L. Eckstein, "The round dataset: A drone dataset of road user trajectories at roundabouts in germany," IEEE, 2020. DOI: 10.1109/ITSC45102.2020.9294728.

[37] A. Breuer, J.-A. Termohlen, S. Homoceanu, and T. Fingscheidt, "Opendd: A large-scale roundabout drone dataset," IEEE, 2020. DOI: 10.1109/ITSC45102.2020.9294301.

[38] T. Moers, L. Vater, R. Krajewski, J. Bock, A. Zlocki, and L. Eckstein, "The exid dataset: A real-world trajectory dataset of highly interactive highway scenarios in germany," in *2022 IEEE Intelligent Vehicles Symposium (IV)*, Piscataway, NJ: IEEE, 2022, pp. 958–964, ISBN: 978-1-6654-8821-1. DOI: 10.1109/IV51971.2022.9827305.

[39] DLR e.V., *Mobile aufbauten*, online.

[40] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The ind dataset: A drone dataset of naturalistic road user trajectories at german intersections," IEEE, 2020. DOI: 10.1109/IV47402.2020.9304839.

[41] S. Fuchs and M. J. Zöllner, *Härtetest für fahrroboter - das testfeld autonomes fahren baden-württemberg wurde in karlsruhe eingeweiht - campusreport am 22.05.2018*, 2018. DOI: 10.5445/DIVA/2018-334.

[42] S. Knake-Langhorst and K. Gimm, "Aim research intersection: Instrument for traffic detection and behavior assessment for a complex urban intersection," *Journal of large-scale research facilities JLSRF*, vol. 2, A65, 2016. DOI: 10.17815/jlsrf-2-122.

[43] H. Weber, J. Bock, J. Klimke, C. Roesener, J. Hiller, R. Krajewski, A. Zlocki, and L. Eckstein, "A framework for definition of logical scenarios for safety assurance of automated driving," *Traffic Injury Prevention*, vol. 20, S65–S70, 2019. DOI: 10.1080/15389588.2019.1630827.

[44] B. Higgs and M. Abbas, "Segmentation and clustering of car-following behavior: Recognition of driving patterns," *IEEE Transactions on Intelligent Transportation Systems*, vol. 16, no. 1, pp. 81–90, 2015, ISSN: 1524-9050. DOI: 10.1109/TITS.2014.2326082.

[45] W. Wang and D. Zhao, "Extracting traffic primitives directly from naturalistically logged data for self-driving applications," *IEEE Robotics and Automation Letters*, vol. 3, no. 2, pp. 1223–1229, 2018. DOI: 10.1109/LRA.2018.2794604.

[46] W. Wang, J. Xi, and D. Zhao, "Driving style analysis using primitive driving patterns with bayesian nonparametric approaches," *IEEE Transactions on Intelligent Transportation Systems*, vol. 20, no. 8, pp. 2986–2998, 2019, ISSN: 1524-9050. DOI: 10.1109/TITS.2018.2870525.

[47] N. de Lange, *Geoinformatics in theory and practice: An integrated approach to geoinformation systems, remote sensing and digital image processing* (Springer textbooks in earth sciences, geography and environment). Berlin and Heidelberg: Springer, 2023, ISBN: 978-3-662-65757-7.

[48] ISO, *Geographic information - referencing by coordinates*, 2019.

[49] R. B. Langley, "The utm grid system," *GPS world*, vol. 9, no. 2, pp. 46–50, 1998.

[50] A. A. Olsen, "Projections and grids," in *Introduction to Digital Navigation*, ser. Springer Series on Naval Architecture, Marine Engineering, Shipbuilding and Shipping, A. A. Olsen, Ed., vol. 16, Cham: Springer Nature Switzerland and Imprint Springer, 2024, pp. 57–100, ISBN: 978-3-031-47489-7. DOI: 10.1007/978-3-031-47490-3_4.

[51] J. Park, D. Lee, and C. Park, "Implementation of vehicle navigation system using gnss, ins, odometer and barometer," *Journal of Positioning, Navigation, and Timing*, vol. 4, no. 3, pp. 141–150, 2015, ISSN: 2288-8187. DOI: 10.11003/JPNT.2015.4.3.141.

[52] Y. Lu, "Discrete frenet frame with application to structural biology and kinematics," Ph.D. dissertation, Florida State University, Florida, 2013.

[53] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenét frame," in *2010 IEEE International Conference on Robotics and Automation*, Piscataway, NJ: IEEE, 2010, pp. 987–993, ISBN: 978-1-4244-5038-1. DOI: 10.1109/ROBOT.2010.5509799.

[54] L. Klitzke, C. Koch, A. Haja, and F. Köster, "Vehicle data management system for scenario-based validation of automated driving functions," in *Smart Cities, Green Technologies and Intelligent Transport Systems*, ser. Communications in Computer and Information Science, M. Helfert, C. Klein, B. Donnellan, and O. Gusikhin, Eds., vol. 1217, Cham: Springer, 2021, pp. 342–362, ISBN: 978-3-030-68028-2. DOI: 10.1007/978-3-030-68028-2_16.

[55] S. Geyer, M. Baltzer, B. Franz, S. Hakuli, M. Kauer, M. Kienle, S. Meier, T. Weissgerber, K. Bengler, R. Bruder, F. Flemisch, and H. Winner, "Concept and development of a unified ontology for generating test and use-case catalogues for assisted and automated vehicle guidance," *IET Intelligent Transport Systems*, vol. 8, no. 3, pp. 183–189, 2014, ISSN: 1751-956X. DOI: 10.1049/iet-its.2012.0188.

[56] S. Ulbrich, T. Menzel, A. Reschka, F. Schuldt, and M. Maurer, "Defining and substantiating the terms scene, situation, and scenario for automated driving," in *2015 IEEE 18th International Conference on Intelligent Transportation Systems*, 2015, pp. 982–988. DOI: 10.1109/ITSC.2015.164.

[57] E. D. Dickmanns, *Dynamic Vision for Perception and Control of Motion*. London: Springer London, 2007, ISBN: 9781846286384. DOI: 10.1007/978-1-84628-638-4.

[58] E. de Gelder, J.-P. Paardekooper, A. K. Saberi, H. Elrofai, O. O. d. Camp, S. Kraines, J. Ploeg, and B. D. Schutter, "Towards an ontology for scenario definition for the assessment of automated vehicles: An object-oriented framework," *IEEE Transactions on Intelligent Vehicles*, vol. 7, no. 2, pp. 300–314, 2022, ISSN: 2379-8858. DOI: 10.1109/TIV.2022.3144803.

[59] G. Bagschik, T. Menzel, and M. Maurer, "Ontology based scene creation for the development of automated vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1813–1820, ISBN: 978-1-5386-4452-2. DOI: 10.1109/IVS.2018.8500632.

[60] F. Schuldt, "Ein beitrag für den methodischen test von automatisierten fahrfunktionen mit hilfe von virtuellen umgebungen," Ph.D. dissertation, 2017. DOI: 10.24355/dbbs.084-201704241210.

[61] L. Hartjen, R. Philipp, F. Schuldt, F. Howar, and B. Friedrich, "Classification of driving maneuvers in urban traffic for parametrization of test scenarios," in *9. Tagung Automatisiertes Fahren*, 2019.

[62] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1821–1827, ISBN: 978-1-5386-4452-2. DOI: 10.1109/IVS.2018.8500406.

[63] H. Weber, C. Glasmacher, M. Schuldes, N. Wagener, and L. Eckstein, "Holistic driving scenario concept for urban traffic," in *IEEE IV 2023*, Piscataway, NJ: IEEE, 2023, pp. 1–8, ISBN: 979-8-3503-4691-6. DOI: 10.1109/IV55152.2023.10186385.

[64] G. Markkula, R. Madigan, D. Nathanael, E. Portouli, Y. M. Lee, A. Dietrich, J. Billington, A. Schieben, and N. Merat, "Defining interactions: A conceptual framework for understanding interactive behaviour in human and automated road traffic," *Theoretical Issues in Ergonomics Science*, vol. 21, no. 6, pp. 728–752, 2020. DOI: 10.1080/1463922X.2020.1736686.

[65] M. Zipfl, N. Koch, and J. M. Zöllner, "A comprehensive review on ontologies for scenario-based testing in the context of autonomous driving," in *IEEE IV 2023*, Piscataway, NJ: IEEE, 2023, pp. 1–7, ISBN: 979-8-3503-4691-6. DOI: 10.1109/IV55152.2023.10186681.

[66] L. Westhofen, C. Neurohr, T. Koopmann, M. Butz, B. Schütt, F. Utesch, B. Neurohr, C. Gutenkunst, and E. Böde, "Criticality metrics for automated driving: A review and suitability analysis of the state of the art," *Archives of Computational Methods in Engineering*, vol. 30, no. 1, pp. 1–35, 2023, ISSN: 1134-3060. DOI: 10.1007/s11831-022-09788-7.

[67] M. Scholtes, L. Westhofen, L. R. Turner, *et al.*, "6-layer model for a structured description and categorization of urban traffic and environment," *IEEE Access*, vol. 9, pp. 59 131–59 147, 2021, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3072739.

[68] M. Althoff, S. Urban, and M. Koschi, "Automatic conversion of road networks from opendrive to lanelets," in *2018 IEEE International Conference on Service Operations and Logistics, and Informatics (SOLI)*, 2018, pp. 157–162. DOI: 10.1109/SOLI.2018.8476801.

[69] Y. Koroglu and F. Wotawa, "Towards a review on simulated adas/ad testing," in *2023 IEEE/ACM International Conference on Automation of Software Test*, A. Garrido, Ed., Piscataway, NJ: IEEE, 2023, pp. 112–122, ISBN: 979-8-3503-2402-0. DOI: 10.1109/AST58925.2023.00015.

[70] M. Haklay and P. Weber, "Openstreetmap: User-generated street maps," *IEEE Pervasive Computing*, vol. 7, no. 4, pp. 12–18, 2008, ISSN: 1536-1268. DOI: 10.1109/MPRV.2008.80.

[71] J. Ziegler, P. Bender, M. Schreiber, *et al.*, "Making bertha drive—an autonomous journey on a historic route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014, ISSN: 1939-1390. DOI: 10.1109/MITS.2014.2306552.

[72] ASAM e.V., *Asam opendrive standard*, 2024.

[73] G. Elghazaly, R. Frank, S. Harvey, and S. Safko, "High-definition maps: Comprehensive survey, challenges, and future perspectives," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 4, pp. 527–550, 2023. DOI: 10.1109/OJITS.2023.3295502.

[74] M. Scholz, *Opendrive dataset of the inner ring road in brunswick*, 2020. DOI: 10.5281/ZENODO.4043192.

[75] C. Wang, Y. Xie, H. Huang, and P. Liu, "A review of surrogate safety measures and their applications in connected and automated vehicles safety modeling," *Accident; analysis and prevention*, vol. 157, p. 106 157, 2021. DOI: 10.1016/j.aap.2021.106157.

[76] J. C. Hayward, *Near miss determination through use of a scale of danger*. 1972.

[77] H. Li, Z. Zhang, N. N. Sze, H. Hu, and H. Ding, "Safety effects of law enforcement cameras at non-signalized crosswalks: A case study in china," *Accident; analysis and prevention*, vol. 156, p. 106 124, 2021. DOI: 10.1016/j.aap.2021.106124.

[78] W. K. M. Alhajyaseen, M. Asano, and H. Nakamura, "Left-turn gap acceptance models considering pedestrian movement characteristics," *Accident; analysis and prevention*, vol. 50, pp. 175–185, 2013. DOI: 10.1016/j.aap.2012.04.006.

[79] J. Nilsson, M. Brannstrom, E. Coelingh, and J. Fredriksson, "Lane change maneuvers for automated vehicles," *IEEE Transactions on Intelligent Transportation Systems*, vol. 18, no. 5, pp. 1087–1096, 2017, ISSN: 1524-9050. DOI: 10.1109/TITS.2016.2597966.

[80] B. L. Allen, B. T. Shin, and P. J. Cooper, "Analysis of traffic conflicts and collisions," *Transportation Research Record*, no. HS-025 846, 1987.

[81] S. Russell and P. Norvig, *Artificial intelligence: A modern approach, global edition*, 4th ed. London, England: Pearson Education, 2021.

[82] C. M. Bishop, *Pattern Recognition and Machine Learning* (Information Science and Statistics). New York: Springer New York, 2016, ISBN: 9781493938438.

[83] L. R. Rabiner, "A tutorial on hidden markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989, ISSN: 0018-9219. DOI: 10.1109/5.18626.

[84] J. A. Bilmes, "A gentle tutorial of the em algorithm and its application to parameter estimation for gaussian mixture and hidden markov models," *International Computer Science Institute*, vol. 4, no. 510, p. 126, 1998.

[85] D. J. Berndt and J. Clifford, "Using dynamic time warping to find patterns in time series," in *Proceedings of the 3rd International Conference on Knowledge Discovery and Data Mining*, ser. AAAIWS'94, AAAI Press, 1994, pp. 359–370.

[86] S. Salvador and P. Chan, "Toward accurate dynamic time warping in linear time and space," *Intelligent Data Analysis*, vol. 11, no. 5, pp. 561–580, 2007, ISSN: 1088467X. DOI: 10.3233/IDA-2007-11508.

[87] L. Klitzke, C. Koch, and F. Köster, "Identification of lane-change maneuvers in real-world drivings with hidden markov model and dynamic time warping," in *2020 IEEE 23rd International Conference on Intelligent Transportation Systems (ITSC)*, 2020, pp. 1–7. DOI: 10.1109/ITSC45102.2020.9294481.

[88] X. Wang, A. Mueen, H. Ding, G. Trajcevski, P. Scheuermann, and E. Keogh, "Experimental comparison of representation methods and distance measures for time series data," *Data mining and knowledge discovery*, vol. 26, no. 2, pp. 275–309, 2013. DOI: 10.1007/s10618-012-0250-5.

[89] L. Klitzke, J. Meyer, T. Leune, C. Koch, and F. Koster, "DAGMaR: A DAG-based robust road membership estimation framework for scenario mining," in *2019 Sixth International Conference on Internet of Things: Systems, Management and Security (IOTSMS)*, M. Alsmirat and Y. Jararweh, Eds., IEEE, 2019, pp. 358–365, ISBN: 978-1-7281-2949-5. DOI: 10.1109/IOTSMS48152.2019.8939213.

[90] D. Powers, "Evaluation: From precision, recall and f-measure to roc, informedness, markedness & correlation," *Journal of Machine Learning Technologies*, vol. 2, no. 1, pp. 37–63, 2011, ISSN: 2229-3981.

[91] J. W. Tukey, "An introduction to the calculation of numerical spectrum analysis," *Spectra Analysis of Time Series*, pp. 25–46, 1967.

[92] FGSV, *Richtlinien für die Anlage von Stadtstraßen: RASt 06*. FGSV Verlag, 2006.

[93] FGSV, *Richtlinien für die Anlage von Autobahnen: RAA*. FGSV Verlag, 2007.

[94] A. A. Hagberg, D. A. Schult, and P. J. Swart, "Exploring network structure, dynamics, and function using networkx," in *Proceedings of the 7th Python in Science Conference*, G. Varoquaux, T. Vaught, and J. Millman, Eds., Pasadena, CA USA, 2008, pp. 11–15.

[95] R. Sedgewick, *Algorithms in C, part 5: graph algorithms*. Pearson Education, 2001.

[96] Z. Huang, S. Qiao, N. Han, C.-a. Yuan, X. Song, and Y. Xiao, "Survey on vehicle map matching techniques," *CAAI Transactions on Intelligence Technology*, vol. 6, no. 1, pp. 55–71, 2021. DOI: 10.1049/cit2.12030.

[97] C.-E. Framing, F.-J. Hasseler, and D. Abel, "Infrastructure-based vehicle maneuver estimation with intersection-specific models," in *2018 26th Mediterranean Conference of Control and Automation (MED 2018)*, Piscataway, NJ: IEEE, 2018, pp. 253–258, ISBN: 978-1-5386-7890-9. DOI: 10.1109/MED.2018.8443069.

[98] C. Rodemerk, H. Winner, and R. Kastner, "Predicting the driver's turn intentions at urban intersections using context-based indicators," in *2015 IEEE Intelligent Vehicles Symposium (IV)*, IEEE / Institute of Electrical and Electronics Engineers Incorporated, 2015, pp. 964–969, ISBN: 978-1-4673-7266-4. DOI: 10.1109/IVS.2015.7225809.

[99] J. Han and M. Kamber, *Data mining: Concepts and techniques*, 3rd ed. Haryana, India and Burlington, MA: Elsevier, 2012, ISBN: 9789380931913.

[100] R. Xu and D. Wunsch, "Survey of clustering algorithms," *IEEE transactions on neural networks*, vol. 16, no. 3, pp. 645–678, 2005, ISSN: 1045-9227. DOI: 10.1109/TNN.2005.845141.

[101] J. H. Ward, "Hierarchical grouping to optimize an objective function," *Journal of the American Statistical Association*, vol. 58, no. 301, pp. 236–244, 1963, ISSN: 0162-1459. DOI: 10.1080/01621459.1963.10500845.

[102] F. Pedregosa, G. Varoquaux, A. Gramfort, *et al.*, "Scikit-learn: Machine learning in python," *Journal of Machine Learning Research*, vol. 12, pp. 2825–2830, 2011.

[103] S. Staab and R. Studer, *Handbook on Ontologies*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, ISBN: 978-3-540-70999-2. DOI: 10.1007/978-3-540-92673-3.

[104] B. N. Grosof, I. Horrocks, R. Volz, and S. Decker, "Description logic programs," in *Proceedings of the 12th international conference on World Wide Web*, G. Hencsey, Ed., ser. ACM Conferences, New York, NY: ACM, 2003, p. 48, ISBN: 1581136803. DOI: 10.1145/775152.775160.

[105] A.-C. N. Ngomo, S. Auer, J. Lehmann, and A. Zaveri, "Introduction to linked data and its lifecycle on the web," in *Reasoning web*, ser. Lecture notes in computer science Information systems and application, incl. Internet/web and HCI, M. Koubarakis, G. Stamou, G. Stoilos, I. Horrocks, P. Kolaitis, G. Lausen, and G. Weikum, Eds., vol. 8714, Cham and Heidelberg: Springer, 2014, pp. 1–99, ISBN: 978-3-319-10586-4. DOI: 10.1007/978-3-319-10587-1_1.

[106] T. R. Gruber, "A translation approach to portable ontology specifications," *Knowledge Acquisition*, vol. 5, no. 2, pp. 199–220, 1993, ISSN: 10428143. DOI: 10.1006/knac.1993.1008.

[107] R. Studer, V. Benjamins, and D. Fensel, "Knowledge engineering: Principles and methods," *Data & Knowledge Engineering*, vol. 25, no. 1-2, pp. 161–197, 1998, ISSN: 0169023X. DOI: 10.1016/S0169-023X(97)00056-6.

[108] M. A. Musen, "The protégé project: A look back and a look forward," *AI matters*, vol. 1, no. 4, pp. 4–12, 2015, ISSN: 2372-3483. DOI: 10.1145/2757001.2757003.

[109] S. Lohmann, S. Negru, F. Haag, and T. Ertl, "Visualizing ontologies with vowl," *Semantic Web*, vol. 7, no. 4, pp. 399–419, 2016. DOI: 10.3233/SW-150200.

[110] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz, "Pellet: A practical owl-dl reasoner," *Journal of Web Semantics*, vol. 5, no. 2, pp. 51–53, 2007, ISSN: 15708268. DOI: 10.1016/j.websem.2007.03.004.

[111] P. Patel-Schneider and M. Horridge, *Owl 2 web ontology language manchester syntax (second edition): W3c note*, 2012.

[112] A. Krisnadhi, F. Maier, and P. Hitzler, "Owl and rules," in *Reasoning Web. Semantic Technologies for the Web of Data: 7th International Summer School 2011, Galway, Ireland, August 23-27, 2011, Tutorial Lectures*, A. Polleres, C. d'Amato, M. Arenas, S. Handschuh, P. Kroner, S. Ossowski, and P. Patel-Schneider, Eds., Berlin, Heidelberg: Springer Berlin Heidelberg, 2011, pp. 382–415, ISBN: 978-3-642-23032-5. DOI: 10.1007/978-3-642-23032-5_7.

[113] J. V. d. Bossche, K. Jordahl, M. Fleischmann, *et al.*, *Geopandas/geopandas*, 2024. DOI: 10.5281/zenodo.10460075.

[114] L. Klitzke, *Xodr2nxgpd - a converter from opendrive to networkx and geopandas*, 2022. DOI: 10.5281/zenodo.7023152.

[115] F. Lange, M. Haiduk, M. Boos, P. Tinschert, A. Schwarze, and F. Eggert, "Road crossing behavior under traffic light conflict: Modulating effects of green light duration and signal congruency," *Accident Analysis & Prevention*, vol. 95, pp. 292–298, 2016, ISSN: 0001-4575. DOI: 10.1016/j.aap.2016.07.023.

[116] J. Wang, H. Huang, P. Xu, S. Xie, and S. C. Wong, "Random parameter probit models to analyze pedestrian red-light violations and injury severity in pedestrian–motor vehicle crashes at signalized crossings," *Journal of Transportation Safety & Security*, vol. 12, no. 6, pp. 818–837, 2020. DOI: 10.1080/19439962.2018.1551257.

[117] J. L. Fleck, C. G. Cassandras, and Y. Geng, "Adaptive quasi-dynamic traffic light control," *IEEE Transactions on Control Systems Technology*, vol. 24, no. 3, pp. 830–842, 2016. DOI: 10.1109/TCST.2015.2468181.

[118] L. Quante, M. Zhang, K. Preuk, and C. Schießl, "Human performance in critical scenarios as a benchmark for highly automated vehicles," *Automotive Innovation*, vol. 4, no. 3, pp. 274–283, 2021, ISSN: 2096-4250. DOI: 10.1007/s42154-021-00152-2.

[119] M. van Steen and A. S. Tanenbaum, *Distributed Systems*, Fourth edition, version 4.01 (January 2023). Maarten van Steen, 2023, ISBN: 9081540637.

[120] G. Bengel, *Grundkurs Verteilte Systeme: Grundlagen und Praxis des Client-Server und Distributed Computing* (Lehrbuch), 4. Aufl. Wiesbaden: Springer Vieweg, 2014, ISBN: 978-3-8348-1670-2. DOI: 10.1007/978-3-8348-2150-8.

[121] G. M. Roy, *RabbitMQ in depth*. Shelter Island: Manning, 2018, ISBN: 9781617291005.

[122] D. Reis, B. Piedade, F. F. Correia, J. P. Dias, and A. Aguiar, "Developing docker and docker-compose specifications: A developers' survey," *IEEE Access*, vol. 10, pp. 2318–2329, 2022, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2021.3137671.

[123] R. T. FIELDING, "Architectural styles and the design of network-based software architectures," Ph.D. dissertation, University of California, Irvine, 2000.

[124] A. Torres, R. Galante, M. S. Pimenta, and A. J. B. Martins, "Twenty years of object-relational mapping: A survey on patterns, solutions, and their implications on application design," *Information and Software Technology*, vol. 82, pp. 1–18, 2017, ISSN: 0950-5849. DOI: 10.1016/j.infsof.2016.09.009.

[125] M. Fowler, *Patterns of enterprise application architecture* (The Addison-Wesley signature series), 17. print. Boston, Mass. and Munich: Addison-Wesley, 2011, ISBN: 9780321127426.

[126] SQLAlchemy, *Sqlalchemy 1.3 documentation: Mixin and custom base classes*, 2021.

[127] L. Klitzke, C. Schicktanz, G. Kay, and M. Henning, *Tasi: A python library for traffic data analysis and situation interpretation*, 2025. DOI: 10.5281/zenodo.17120267.

[128] B. Ivanovic, G. Song, I. Gilitschenski, and M. Pavone, *Trajdata: A unified interface to multiple human trajectory datasets*, 2023.

[129] A. Paszke, S. Gross, F. Massa, *et al.*, "Pytorch: An imperative style, high-performance deep learning library," in *Advances in Neural Information Processing Systems*, H. Wallach, H. Larochelle, A. Beygelzimer, F. d\textquotesingle Alché-Buc, E. Fox, and R. Garnett, Eds., vol. 32, Curran Associates, Inc, 2019.

[130] T. A. Caswell, E. S. de Andrade, A. Lee, *et al.*, *Matplotlib/matplotlib: Rel: V3.7.5*, 2024. DOI: 10.5281/ZENODO.592536.

[131] D. Vohra, Ed., *Practical Hadoop ecosystem: A definitive guide to Hadoop-related frameworks and tools* (For professionals by professionals). New York, NY: Apress, 2016, ISBN: 978-1-4842-2198-3. DOI: 10.1007/978-1-4842-2199-0.

[132] W. McKinney *et al.*, "Pandas: A foundational python library for data analysis and statistics," *Python for high performance and scientific computing*, vol. 14, no. 9, pp. 1–9, 2011.

[133] GEOS contributors, *Geos coordinate transformation software library*, 2021.

[134] P. Songchitruksa and A. P. Tarko, "The extreme value theory approach to safety estimation," *Accident Analysis & Prevention*, vol. 38, no. 4, pp. 811–822, 2006, ISSN: 0001-4575. DOI: 10.1016/j.aap.2006.02.003.

[135] I. I. Da Silva, M. Zhang, and K. Gimm, "A collaborative framework for semi-automatic scenario-based mining of big road user data," in *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2023, pp. 5100–5105, ISBN: 979-8-3503-9946-2. DOI: 10.1109/ITSC57777.2023.10422330.

[136] J. H. Kell, I. J. Fullerton, and M. K. Mills, *Traffic detector handbook. second edition*, 1990.

[137] C. Leschik, M. Zhang, and K. Gimm, "Analysis of stopping behaviour of cyclists at a traffic light-controlled intersection using trajectory data," *The Evolving Scholar*, 2023. DOI: 10.24404/63FE36F6FD33480B61C11947.

[138] C. Bouvie, J. Scharcanski, P. Barcellos, and F. L. Escouto, "Tracking and counting vehicles in traffic video sequences using particle filtering," in *2013 IEEE International Instrumentation and Measurement Technology Conference (I2MTC 2013)*, Piscataway, NJ: IEEE, 2013, pp. 812–815, ISBN: 978-1-4673-4621-4. DOI: 10.1109/I2MTC.2013.6555527.

[139] J. Zheng, X. Ma, Y.-J. Wu, and Y. Wang, "Measuring signalized intersection performance in real-time with traffic sensors," *Journal of Intelligent Transportation Systems*, vol. 17, no. 4, pp. 304–316, 2013, ISSN: 1547-2450. DOI: 10.1080/15472450.2013.771105.

[140] M. Gramaglia, C. J. Bernardos, and M. Calderon, "Virtual induction loops based on cooperative vehicular communications," *Sensors (Basel, Switzerland)*, vol. 13, no. 2, pp. 1467–1476, 2013. DOI: 10.3390/s130201467.

[141] S. Knake-Langhorst, "Generische systemarchitektur für die erhebung mikroskopischer verkehrsdaten," Ph.D. dissertation, Technischen Universität Berlin, Berlin, 2022. DOI: 10.14279/DEPOSITONCE-15363.

[142] L. Klitzke, "Infrastructurally collected traffic data for scenario extraction and analysis," in *VVM final event*, 2023.

[143] C. Schicktanz, L. Klitzke, and K. Gimm, "Microscopic analysis of the impact of congestion on traffic safety and efficiency at a signalized intersection: A case study," in *2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2023, pp. 2827–2834, ISBN: 979-8-3503-9946-2. DOI: 10.1109/ITSC57777.2023.10422205.

[144] L. Klitzke, C. Leschik, and K. Gimm, "Investigation on road traffic safety in rural areas using trajectory data: Case studies at two measurement sites," *Traffic Safety Research*, 2025. DOI: 10.55329/ditw1500.

[145] A. Rasch, S. Moll, G. López, A. García, and M. Dozza, "Drivers' and cyclists' safety perceptions in overtaking maneuvers," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 84, pp. 165–176, 2022, ISSN: 13698478. DOI: 10.1016/j.trf.2021.11.014.

[146] G. F. Bianchi Piccinini, C. Moretto, H. Zhou, and M. Itoh, "Influence of oncoming traffic on drivers' overtaking of cyclists," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 59, pp. 378–388, 2018, ISSN: 13698478. DOI: 10.1016/j.trf.2018.09.009.

[147] L. Klitzke, K. Gimm, C. Koch, and F. Köster, "Extraction and analysis of highway on-ramp merging scenarios from naturalistic trajectory data," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, 2022, pp. 654–660, ISBN: 978-1-6654-6880-0. DOI: 10.1109/ITSC55140.2022.9922191.

[148] H. Jula, E. B. Kosmatopoulos, and P. A. Ioannou, "Collision avoidance analysis for lane changing and merging," *IEEE Transactions on Vehicular Technology*, vol. 49, no. 6, pp. 2295–2308, 2000, ISSN: 00189545. DOI: 10.1109/25.901899.

[149] W. Qi, W. Wang, B. Shen, and J. Wu, "A modified post encroachment time model of urban road merging area based on lane-change characteristics," *IEEE Access*, vol. 8, pp. 72 835–72 846, 2020, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2020.2987959.

[150] X. Cui, X. Li, X. Zheng, and Y. Ren, "Driving behavior primitive classification using cnn-based fusion models," *IEEE Access*, vol. 12, pp. 56 344–56 355, 2024, ISSN: 2169-3536. DOI: 10.1109/ACCESS.2024.3391170.

[151] L. Westhofen, C. Neurohr, J. C. Jung, and D. Neider, "Answering temporal conjunctive queries over description logic ontologies for situation recognition in complex operational domains," in *Tools and Algorithms for the Construction and Analysis of Systems*, ser. Lecture Notes in Computer Science, B. Finkbeiner and L. Kovács, Eds., vol. 14570, Cham: Springer Nature Switzerland and Imprint Springer, 2024, pp. 167–187, ISBN: 978-3-031-57245-6. DOI: 10.1007/978-3-031-57246-3_10.

[152] L. Westhofen, C. Neurohr, M. Butz, M. Scholtes, and M. Schuldes, "Using ontologies for the formalization and recognition of criticality for automated driving," *IEEE Open Journal of Intelligent Transportation Systems*, vol. 3, pp. 519–538, 2022. DOI: 10.1109/OJITS.2022.3187247.