KI WISSEN
Automotive AI Powered by Knowledge

Conjoint Whitepaper
Deliverable D1 & D4

Dominik Grundt[15], Philipp Borchers[15], Laura von Rueden[13], Hendrik Königshof[14], Stefan Griesche[16], Aiman Hsino[5], Daniel Bär[6], Etienne Bührle[14], Han Chen[5], Tobias Gleißner[12], Philip Gottschall[12], Maximilian Grabowski[3], Florian Grötzner[11], Sebastian Houben[13], Niklas Keil[1], Johann Kelsch[9], Tobias Latka[10], Denny Mattern[12], Stefan Matthes[11], Artem Oppermann[4], Benjamin Wahl[4], Adrian Paschke[12], Claus Pastor[3], Syed Tahseen Raza Rizvi[8], Jörg Reichardt[6], Stefan Rudolph[10], Gerhard Schunk[18], Puria Sheikhipour[17], Gurucharan Srinivas[9], Hendrik Stapelbroek[5], Vera Stehr[18], Anh Tuan Tran [16], Abhishek Vivekanandan[14], Florian Wasserrab[1], Pablo Zuazo[1], Stefan Zwicklbauer[6]

[1]Alexander Thamm GmbH
[2]AVL Software and Functions GmbH
[3]Bundesanstalt für Straßenwesen (BASt)
[4]BTC Embedded Systems AG
[5]Capgemini Engineering
[6]Continental AG
[8]Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI)
[9]Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)
[10]Elektronische Fahrwerksysteme GmbH (EFS)
[11]fortiss GmbH
[12]Fraunhofer-Institut für offene Kommunikationssyteme (FOKUS)
[13]Fraunhofer-Institut für Intelligente Analyse- und Informationssyteme (IAIS)
[14]FZI Forschungszentrum Informatik (MPS & TKS)
[15]OFFIS e.V.
[16]Robert Bosch GmbH
[17]Universität des Saarlandes (UdS)
[18]Valeo Schalter und Sensoren GmbH

Version	Date	Comment
1.0	01/01/2022	First version

Gefördert durch:

Bundesministerium für Wirtschaft und Energie

aufgrund eines Beschlusses des Deutschen Bundestages

CONTENTS

1 PROJECT INTRODUCTION

The research project KI Wissen develops methods for integrating existing knowledge into the data-driven AI functions of autonomous vehicles. The goal of the project is to create a comprehensive ecosystem for the integration of knowledge into the training and safeguarding of AI functions.

Up until now, the development of AI functions has been purely driven by data. However, this data-driven approach requires enormous amounts of data for the training and validation of AI functions, with the collection and processing of this data being very resource-intensive and expensive. In addition to the dependence on extensive amounts of data, data-based AI processes have another weakness: they are still generally black-box models for which the decision-making process cannot be directly reconstructed. Previous research approaches to solving these issues have focused on optimising the data needed to train and validate AI functions.

The research project KI Wissen addresses the challenge in a novel way and develops methods and tools for integrating knowledge into machine learning. The project consortium, consisting of 16 partners from industry, research and science under the leadership of Continental, investigates how existing, traffic-relevant knowledge - in the form of traffic rules, mathematical-physical conditions and also social norms - can be integrated into the data-driven AI functions of autonomous vehicles. By combining conventional data-based AI methods with the knowledge- or rule-based methods developed in the project, the basis for the training and validation of AI functions is largely redefined. This now includes not only data, but knowledge, i.e. data, their relations and information. The further development carried out in the project addresses the central challenges on the way to autonomous driving: the generalisation of AI to phenomena with a small data basis, the increase in the stability of the trained AI to disturbances in the data, the data efficiency, the plausibility and the validation of AI-supported functions as well as the increase in functional quality.

A broad variety of methods will be developed and investigated in KI Wissen: methods for knowledge integration, knowledge extraction and knowledge conformity. The methods developed in the project will be evaluated and demonstrated using three defined use cases. Specifically, the following core scientific and technical innovations will be addressed:

Scientific innovations:

- Development of approaches to identify and formalise relevant domain knowledge for L3 to L5 driving functions;
- Development of evaluated approaches for knowledge-based training of AI components;
- Development of learning techniques for training reduction and performance improvement;
- Creation of an ecosystem for knowledge as a basis for efficient training and the safeguarding of AI components based on formalised knowledge.

Technical innovations:

- Development of reusable qualified and formalised modules for application, safeguarding, and network knowledge (e.g., formalised road traffic regulations with exceptions);
- Creation of a tool kit of formalised knowledge and corresponding AI methods for companies and research;
- Creation of demonstrators with knowledge-based AI components.

2 Overview of Deliverables

The research project KI Wissen will work on the set goals over a period of three years. It pursues the step-by-step improvement of the methods to be developed in three successive project steps, the project increments. At the end of each project increment, which goes hand in hand with the project milestones, the (interim) results achieved in the work packages are documented in the form of deliverables. The main objective of the deliverable documents is to communicate the achieved results within the project and also beyond it, towards the involved stakeholders, funding body, and other research activities. There are in total seven deliverables which are defined in KI Wissen. In terms of content, they are oriented towards the technical and scientific project innovations:

ID	Title	Release	Type
D1	Catalog, characterization, and representation of relevant domain knowledge for L3 to L5 driving functions.	public	Document
D2	Learning techniques for integrating formalized knowledge and network knowledge for training reduction and performance improvement	public	Document/ White paper
D3	Methods for extracting knowledge from a data-driven AI function	public	Document
D4	Methods for data-independent validation of the predictions and decisions of an AI function	public	Document
D5	Reusable, qualified and formalized modules of application, safeguarding and network knowledge	VDA LI	Software and Specifications
D6	Construction kit of knowledge-based AI methods for companies and research	VDA LI	Software and specifications
D7	Demonstrators with knowledge-based AI components	project-internal	Software and Specifications/ Demonstration

TABLE 1: Overview of the deliverables

3 PROJECT BACKGROUND

In this section we provide additional project information via the scope of the involved subprojects in subsection 3.1 and the use cases in subsection 3.2

3.1 Contributing Subprojects

Subproject 1 (SP1)

In SP1, various methods for integrating knowledge into AI functions are developed and investigated. This is followed by the integration of knowledge into training procedures of existing systems and the development of new architectures incorporating knowledge for applications in autonomous driving. The main objective is to reduce the amount of data, needed for training, or the computing power required, while at the same time increasing the generalization capability or functional quality of AI functions.

For the use of domain and expert knowledge in data-driven AI methods, this subproject provides with WP1.5 a catalog of standardized knowledge representations. This includes both the aspect of immutable physical/mathematical knowledge from WP1.3 and the aspect of negotiable norms and rules from WP1.4. Both aspects are necessary prerequisites for the usability, exchangeability, and reusability of knowledge in training, validation, and assurance of data-driven methods in SP1, SP2 and SP3 as well as the integration capability into the demonstrator.

Work package 1.3 (WP1.3) - Physical and mathematical laws

In WP1.3, the methods developed in WP1.1 and WP1.2 for taking existing knowledge into account are to be used to identify relevant physical and mathematical knowledge for each scenario and to incorporate it explicitly into the AI model. After a suitable formalization of this knowledge, the learning process or the architecture of the AI model is enriched so that its predictions become compatible with these laws and thus make it able to generalize and increase data efficiency. The developed procedures should make it possible, through the active use of the formalized laws in the training process, to reduce the amount of labeled data significantly. This should not only reduce the effort and costs involved in the acquisition and annotation of the data, but also make it possible to use a large proportion of synthetic data, which can be obtained much more cheaply, for training. The results in WP1.3 should contribute to all three use cases.

Work package 1.4 (WP1.4) - World and expert knowledge

The work package WP1.4 concentrates on identifying relevant world and expert knowledge for autonomous driving and transforming this knowledge into a machine readable knowledge representation. While today's maneuver planners for instance are limited to simple and hard-coded norms, this AP will also work on the more complex norm structures. In particular this covers unwritten norms as well as exceptions and the resolution of norm conflicts, for example in the form of a "controlled rule-exception" of a written norm in favor of a higher-ranking rule or legal paragraph.

For the implementation, novel approaches of neural-symbolic knowledge representation for the integration of heterogeneous semantic knowledge and logical reasoning into the learning methods are suitable. This includes the transformation from textual data into an intermediate representation such as LegalRuleML or text embeddings. Those intermediate representation can than be further processed to formulate logical rules. Other approaches deal with ontologies to formalize expert knowledge and apply the modeled relation and concepts in a knowledge graph embedding for further reasoning (for example in a perception task).

Work package 1.5 (WP1.5)

This work package serves the project-wide connection of the models and forms of knowledge representation developed in the individual subprojects. Through the structured processing of the different forms of knowledge, these methods can be used beyond the scope of the project in the entire AI project family and in other projects beyond the automotive context. Accordingly, the partners involved in this subproject develop methods for knowledge representation, transfer knowledge into such representations, or integrate corresponding representations into AI methods. In order to automatically acquire knowledge and integrate it into

learning systems, it should be represented in a compatible format. In this project, knowledge from various sources (legal and social norms, physics, understanding of the world, etc.) is used. Furthermore, the types of knowledge used differ significantly in terms of their generalizability. For example, the mathematical-physical principles considered in WP1.3 form inviolable knowledge, while the traffic rules or social norms of behavior examined in WP1.4 are only followed by most of the road users. To ensure the high safety requirements, compliance with essential safety rules by the AI system must be verifiable and any exceptions to individual traffic rules must be reliably identified. Rule violations by other road users, on the other hand, are to be expected sporadically and must be mitigated as far as possible by adequate own behavior. According to the fundamentally different sources and meaning, knowledge is represented in different forms, ranging from (differential) equations for physical knowledge and ontologies to graph-based probabilistic logic networks and neural networks that are difficult for humans to comprehend.

Subproject 3 (SP3)

One of the main goals in SP3 is to discover decisions of an AI that are nonconforming to formalized knowledge. This will be achieved on the basis of physical and mathematical knowledge (WP3.1) and world and expert knowledge (WP3.2). The methods to be developed in SP3 will enable testing of AIs based on knowledge so that they can be further trained in a targeted manner (handover to SP1). Furthermore, the testing of knowledge conformance at runtime in the vehicle will be enabled, so that safety-relevant wrong decisions can be detected and avoided. In addition to detecting inconsistencies between the output of AI components and existing knowledge, this should improve the robustness and operational safety of AI components. The ability to match AI predictions with existing knowledge will also increase the confidence of AI components as well as improve the explainability of AI predictions. By integrating AI components with traditional methods (hybrid architectures), a reduction of the training effort is to be achieved in order to meet requirements for the runtime of AI training.

Work package 3.1 (WP3.1)

In WP3.1, methods of algorithmic inference about physical phenomena [1] will first be extended to test mechanisms for AI components. These will be used both in the running operation of an AI component as a monitor (e.g., for the use of external concepts such as the activation of safe fallback levels) and for the control of learning-on-demand (i.e., the selection of previously unlabeled training data controlled by the learning AI) in cooperation with SP1.

Topics are on the one hand the detection of inadmissible generalization for the purpose of error masking as well as hybrid AI and safety architectures (which e.g. mix different AI-based components with non-AI-based components) in order to obtain a plausibility check of the models on a functional level. Further, the use of knowledge about mathematical and physical regularities will be used to evaluate network outputs, to evaluate the predictions of the DNN when the input data is deliberately manipulated, and to enable out-of-context detections with it.

The knowledge formalizations used in WP3.1 are based on the results of SP1 and SP2, in particular the networked knowledge representations developed in WP1.5. Use cases 1 and 2 are considered in WP3.1.

The evaluation criteria and architectures developed in WP3.1 are intended to capture mathematical and physical inconsistencies with given regularities in real time. The global validity of physical laws of nature and mathematical regularities enables a cross-domain use of the quantitatively formalized knowledge building blocks. Furthermore, if the conformity is checked positively, the developed methods offer the possibility to better understand the AI models and to plausibilize their decisions, also for the general public. In case of negative verification, the detected inconsistencies can be returned to SP1 for further targeted training and improvement of the AI.

Work package 3.2 (WP3.2)

For the outputs of AI-based components in autonomous driving to be reliable, robust, and comprehensible, they should conform to world and expert knowledge. World knowledge generally describes knowledge that every human normally

has about the general environment. This world knowledge may be intuitive, e.g., a human cannot walk on water, or a car cannot fly. Expert knowledge describes knowledge that a certain group of people possesses. In the context of driving, this can be knowledge about the specific environment or region, e.g., that there are pedestrian paths next to the current road, that cars drive on the left side in Great Britain, or that certain behavior complies with traffic norms. Such world and expert knowledge should be reflected in the outputs of AI-baased procedures.

A first challenge for checking conformity to world and expert knowledge in autonomous driving is the prior identification of relevant knowledge sources and their representation. Since these knowledge categories often describe intuitive knowledge, an explicit step for formalization is often necessary. Depending on the formalization, different possibilities arise for procedures to make the knowledge usable during training or to check the outputs of the AI components.

In WP3.2, sources of world and expert knowledge that can be used to validate the knowledge conformance of AI-based components in autonomous driving are identified, and methods are developed to perform this validation. The knowledge sources include both spatial knowledge, such as geographic and visual concepts, and traffic knowledge, such as social and legal traffic norms. The solution approach includes the development of methods to apply the knowledge to the AI-based components for matching and validation of the learned models, as well as already in the vehicle at runtime. The methods developed in WP3.2 allow a playback to the knowledge integration methods developed in SP1, as well as a demonstration in SP4. Since world and expert knowledge is available in comprehensive areas and can be used in many ways, this work package can in principle contribute to all use cases.

The approaches developed in WP3.2 allow an evaluation of the qualitative goodness of AI predictions based on world and expert knowledge. The developed evaluation metrics enable the detection and thus the avoidance of safety-relevant wrong decisions. This makes the AI components used in autonomous driving more reliable, more robust and more comprehensible.

3.2 Use Cases

To demonstrate the capabilities during the project and test their improvement on handling critical traffic scenarios, the consortium defined three Use Cases before project start. These Use Cases covers the three stages perception, situation recognition and planning, where knowledge can be applied on. The three Use Cases are:

- UC1: Pedestrian Detection
- UC2: Lane Change
- UC3: Intended and Controlled Rule Exception

These Use Cases are described in Table 2.

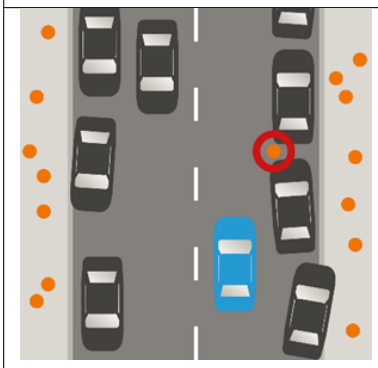
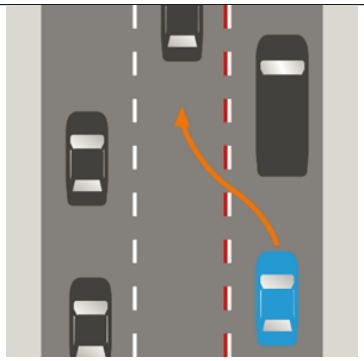
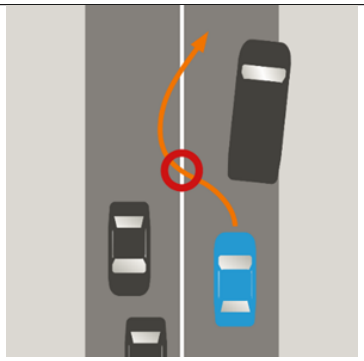
UC1: Pedestrian Detection	**UC2: Lane Change**	**UC3: Controlled Rule Exception**
		
Description: In this Use Case an automated driving (AD) vehicle (blue car) is driving on a street with densely parking vehicles near to pedestrian's sidewalks. The pedestrians that are walking on these sidewalks can be only detected partially by the AD vehicle. In a critical situation one pedestrian is occurring between two parking vehicles to cross the street.	**Description:** In this Use Case a AD vehicle (blue car) is driving on a street multiple lanes. The AD vehicle performs a lane change and has to be aware of other not explicitly specified vehicles and their behavior.	**Description:** In this Use Case a AD vehicle (blue car) is driving on street with one lane per direction with a continuous white line between the lanes. Somewhere in the front there is an obstacle blocking the lane, such that the AD vehicle could not drive ahead without crossing the white line in the middle. Additionally there can be oncoming traffic.
Intention: This Use Case is intended to demonstrate the benefit of an improved pedestrian detection. If pedestrians can be detected at an early stage an their behavior can be predicted, the vehicle can react in the right time and avoid a collision.	**Intention:** In order to reduce the risk of congestion and accidents caused by a lane change, traffic participants should behave cooperatively. This cooperation requires a pre detection of a lane change. Additionally it benefits from a prediction of other traffic participants behavior. Both, detection and prediction, can be demonstrated in this use case.	**Intention:** In general, the AD vehicle is not allowed to cross the continuous white line by traffic rules. This Use Case is intended to demonstrate, that the AD vehicle can detect the special lane blocking situation and decide to break the traffic rule under safe conditions.

TABLE 2: Description of the three Use Cases in KI-Wissen: Pedestrian Detection (UC1), Lane Change (UC2) & Controlled Rule Exception (UC3).

4 First Concepts

The work packages 1.3, 1.4, 1.5, 3.1 & 3.2 in the KI Wissen project deal with integration and conformity check of mathematical & physical knowledge as well as world & expert knowledge. Regardless of whether knowledge is integrated or it's conformity is checked, the source, formalization and representation of the knowledge are of interest. Hence the project partners in those APs have formed a focus group knowledge sources to collected their approaches and concepts regarding knowledge formalization, representation, integration and conformity check from the first project year. The results are documented in this section. In the following tables we present an overview of the individual partner approaches. The work is classified via the four categories:

1) Use Cases
2) Knowledge Sources
3) Knowledge Formalization & Representation
4) Conformity Check

The use cases are the pre-defined KI-Wissen project use cases *Pedestrian detection* (UC1), *Lane Change* (UC2), and *Rule Exception* (UC3) described in subsection 3.2. The classification in the other three categories is inspired by the informed machine learning taxonomy from [2, 3]. The knowledge source describes the type or domain of the knowledge. The knowledge representation & formalization describe the abstract or mathematical format of the knowledge. The conformity check category describes the concept that is used for validating AI models with the knowledge. We identified the classes for each of the knowledge-related categories in the tables below by first assigning key words to every partner approach, and then clustering them (e.g. knowledge on evolution, traffic rules, etc.).

Use Cases		
UC 1: Pedestrian Detection	**UC 2: Lane Change**	**UC 3: Rule Exception**
• Alexander Thamm • AVL • BASt • BTC-ES • Capgemini • FhG IAIS • OFFIS	• BASt • BTC-ES • Capgemini • Continental • DFKI • DLR • EFS • fortiss • FZI MPS • FZI TKS • OFFIS • Robert Bosch • UdS • Valeo	• BASt • Continental • FhG FOKUS • FZI MPS • FZI TKS • Robert Bosch • UdS

Knowledge Sources				
Knowledge on Evolution	**Knowledge on Scene**	**Traffic Rules & Requirements**	**Spatial Knowledge**	**Social Norms**
• AVL • BTC-ES • Continental • EFS • FhG FOKUS • FZI TKS • OFFIS	• BTC-ES • DLR • FhG FOKUS • FZI TKS • fortiss • Continental • OFFIS • Valeo	• BASt • Continental • BASt • fortiss • FhG FOKUS • FZI MPS • Robert Bosch • UdS	• Alexander Thamm • AVL • Capgemini • DFKI • DLR • FhG IAIS • Valeo	• FZI MPS

Knowledge Formalization & Representation					
Equations & Inequalities	**Logic**	**Knowledge Graphs**	**Intermediate Network Representation**	**Visual Representation**	**Mesh grid, Point cloud**
• AVL • Continental • EFS • fortiss	• BTC-ES • FhG FOCUS • FZI MPS • OFFIS • UdS	• BASt • Continental • DFKI • DLR • FhG IAIS • FZI MPS • Robert Bosch	• Continental • Robert Bosch	• Capgemini • FhG IAIS • FZI TKS • Valeo	• Valeo

Conformity Check					
Verifying Semantic Detection Region	**Monitoring**	**Causal Queries & Sensitivity Analysis**	**Anomaly Detection**	**Conformity Metrics**	**Equivariance Constraints & Likelihoods**
• Capgemini	• Alexander Thamm • AVL • BTC-ES • Capgemini • FhG FOCUS • fortiss • FZI MPS • FZI TKS • OFFIS	• EFS	• Alexander Thamm • AVL • FhG IAIS • Valeo	• FhG IAIS • UdS	• Continental

4.1 Alexander Thamm GmbH

Topic overview:

Use Cases (Scenario)
UC1

Knowledge Source
2d object distributions, intermediate network representations

Knowledge Formalization & Representation
No involvement

Conformity Check
Deviations from extracted characteristics, evaluation on training support

4.1.1 Introduction

Since we as Alexander Thamm GmbH want to rather concentrate on conformity checks instead of putting effort to data integration, we focus on approaches that do not require external knowledge sources. The basic idea is to extract certain concepts or regularities of objects of interest in traffic scenes from the training data. The extracted concepts serve as a knowledge source and can subsequently be used to check individual predictions of the model with respect to their conformity. For this purpose, several approaches are to be implemented and combined in order to ultimately provide a well-founded conformity check. The output of this method can then provide, for example, indications that the prediction in the current traffic situation is not compliant with the training data which could imply that the test instance is out-of-distribution and thus cannot be predicted with high reliability. This conformity value can be used to issue a warning in such situations or to develop more complex control mechanisms based on it. Additionally, the information of the extracted concepts could also be used to improve the performance of the model. This can be achieved by integrating these extracted concepts into the model itself or by using this information to restructure the train/validation split of the input samples with a subsequent re-training of the model.

4.1.2 Use Cases

We will mainly focus on Use Case 1, where the approaches described here can in principle be applied to all sub-use cases. In the future, application to Use Case 2 might also be conceivable by applying the concepts not only to the raw image data of the ego vehicle, but also on the bird's eye view of the traffic scene. Here, the idea would also be to evaluate conformity of the model decision with known traffic situations by comparing against the training data.

4.1.3 Knowledge sources

Extracted concepts/regularities from the training data:

- 2d distributions of pedestrians with respect to their localization in the image
- Extracted intermediate representations of classification/object recognition models with respect to pedestrians (either individually or also as a distribution/mean/etc.)

4.1.4 Representation/Formalization concept

We are not involved in the integration and formalization of knowledge.

4.1.5 Conformity check

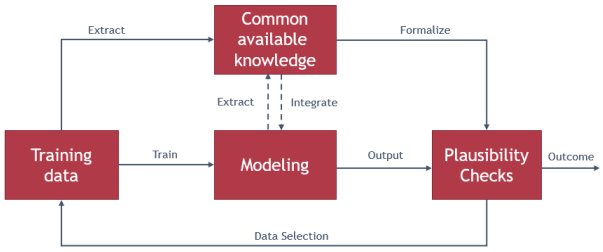


Fig. 1: General approach of using extracted knowledge for conformity checks

In order to implement the basic idea described in the introduction, the first step is to extract the concepts from the training data. Depending on the method, this is done either before or during the training of the model. Afterwards, the extracted knowledge has to be formalized in a way that it can be used for further computations. In the end, the extracted knowledge is then used as a comparison basis of the conformity check to identify major deviations of the model prediction from the training data, which is then output in the form of

a warning or similar when the score surpasses a threshold. Additionally, these scores can also be directly integrated into the prediction process of the AI model itself or used for specific data selection to strengthen underrepresented or edge cases in the training data. To make the final conformity score more reliable, it is planned to combine multiple of them based on different approaches.

Object location-based approach

Keeping the general idea in mind, the first approach deals with using the positions of the pedestrians in the training data to compute a two-dimensional distribution. The computation of the distribution is done in a model-independent manner as it only requires the object annotations of the dataset and serves as a possible formalization of the knowledge about pedestrians extracted from the training data. To do so, as a first step all object annotations of the training set must be loaded and filtered by the object class of interest, which is "pedestrian" in this case. If the dataset contains multiple camera orientations of the ego vehicle (e.g. front, left, right and back), one additionally has to group by the individual cameras, since the positions of objects vary significantly depending on the point of view and thus can not be mixed or compared. Note that the approach works with every object class and camera orientation. Next, having only the object annotations of pedestrians remaining, the center coordinates of the bounding boxes are computed and stored in a list. Scatter plotting this gives an empirical overview of where pedestrians in the training set were located.
Based on the empirical distribution, a Gaussian kernel density is then being estimated to best represent the pedestrian occurrences in the training data while also generalizing well. To achieve this, it might be helpful to do a hyper parameter optimization to find an appropriate probability density function (pdf). The resulting pdf then also reflects physical properties like the fact that it is very unlikely to see a person somewhere up in the sky.
Finally, if the model detects a pedestrian in any unseen traffic situation, the density function can then be used to calculate a probability score that expresses how compliant the model prediction is with pedestrians from previous training data. For this, the center coordinates of the detected pedestrian have to be computed and are subsequently fed into the density function. Since the values can get quite small, it might be preferable to transform them in some way, like using the log values instead.

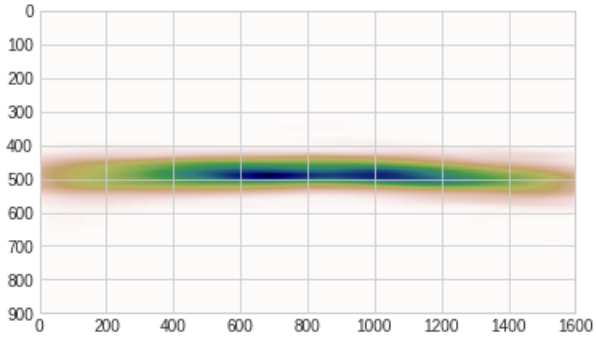


Fig. 2: Gaussian kernel density estimation of pedestrian locations on a random subset of the front camera of the nuImages dataset

An open challenge is to find a threshold above which predictions are then classified as conform or not conform. As this depends on multiple factors like the camera view or hyper parameters of the kernel density estimation, this has to be evaluated on the validation set individually. A helpful metric for this purpose can be the ratio of detected pedestrians that were correctly classified as not conform, i.e. the prediction gets rejected and there is indeed no object at that location in the ground truth.
Since this method is straight forward and comparatively easy to implement, it was decided to test it first and see whether this already achieves some helpful results. As potential further elaboration, one could transfer this approach to sensor data like lidar and radar and estimate a density function in a similar manner, but this time from a bird's eye view.

Deep k-Nearest Neighbours

Another method to be evaluated in the context of autonomous driving is a modification of the Deep k-Nearest Neighbours (DkNN) [4]. Originally,

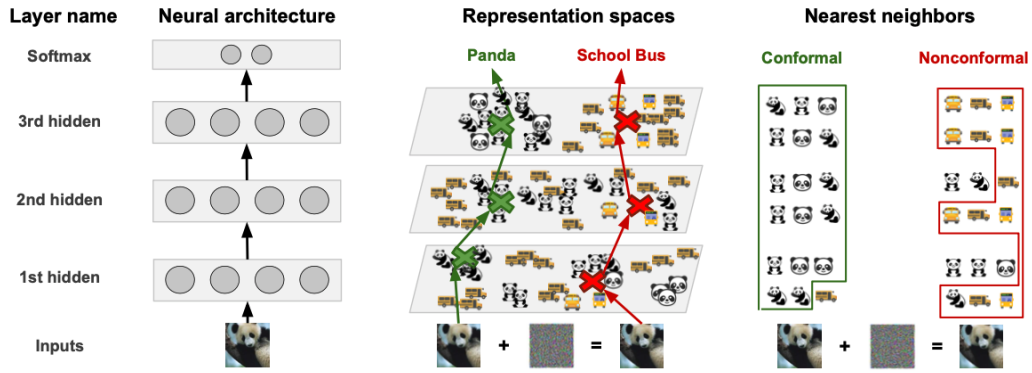


Fig. 3: Exemplary classification task when the input image is modified [4]

this approach involves training a neural network on a classification task in advance. Subsequently, the intermediate representations of the individual training instances that are created during training are stored and form the basis for constructing conformity checks. If the predictor is then applied to unseen data, the input is processed by the neural network and the differences from resulting intermediate representations to the intermediate representations from the training process are calculated.

As in a normal kNN model, however, only the k nearest neighbours are used for comparison. Depending on the specific approach and the architecture of the neural network, several additional indicators could be obtained (e.g. multiple representations corresponding to the number of hidden layers). The calculated differences can provide an intuition of how conform the model output is with the extracted knowledge from the training data. In the case of very large deviations from the neighbours, situations can thus be detected that are not covered by the training data, which in turn indicates that the prediction of the model is not reliable. The principle of this method is illustrated in figure 3 using the example of the classification of pandas and buses. Here it is shown, for example, that an image combined with a noise (i.e. "adversarial attack") as input leads to non-homogeneous nearest neighbours leading to a prediction with high uncertainty.

We now want to apply this underlying idea to object detection, e.g. recognizing possible pedestrians in traffic situations. Since popular object detection models like R-CNNs rely on a CNN as feature extractor, we can also compare intermediate representations in this context and produce a conformity check based on it. One aspect that needs to be taken into account here is the potential computational overhead if many promising regions defined by a region proposal network are evaluated by comparing their intermediate representations.

Uncertainty metrics

There are alternative metrics that can be constructed on top of these intermediate representations of input samples. For instance, surprise adequacy [5] has been used to measure the relative novelty of unseen data compared with all data used during the training phase, by comparing the activations of a new input against the activations of all training inputs. For a given new input, each targeted activation value is compared against those activations observed during training (for all training inputs or for those from the predicted class only), which can be done for instance using the mean Euclidian distance or a probability density estimation. This serves to recognize intrinsic differences of new inputs with respect to the training data. Yet another simple but powerful alternative to identify out-of-distribution inputs (that will potentially be misclassified) are several uncertainty metrics [6] based on the final probabilities of a deep learning system. Here, the prediction probabilities of the model

and the variance of these probabilities (obtained from multiple predictions on the same individual input by using the same trained model where random dropouts are applied) are used alone or in combination to assess the model uncertainty.

As for the previous cases, incorporating the information extracted by these metrics into the neural model serves as a conformity check of the intrinsic characteristics of the input data treated as a whole, which can also improve the performance of the network. More in particular, the previously described metrics can also be used for the selection of train/test input data, which was proved to be an efficient approach to improve model performance. This approach is based on identifying input cases that are likely to trigger a misclassification (e.g. out-of-distribution samples), as these uncertain cases seem to be the most informative ones for guiding the training process. Using the out-of-distribution inputs in the training set allows for a higher model accuracy after training compared to random selection of training inputs.

Alternative approaches

During the research phase, several approaches different to above mentioned ones were considered as well. Depending on the initial results and the time required for the other approaches, additional methods for conformity checks might be evaluated in the context of the project.

One of them also fits to our general idea and does not require the inclusion of external knowledge. In this case, prior color knowledge is extracted from the objects and background scenes and can either be used to influence or assess the model decision. A simple way to implement this would, for instance, be to calculate the mean image or mean color ratio of all bounding boxes (i.e. the cut out objects) or certain masked areas (e.g. the area where road lanes usually appear) of the same class. After inference, one could then calculate the difference, e.g. Euclidean distance, of the detected bounding box or masked area to the means of each class and thus obtain an indicator of how conform the prediction is to the prior-known color knowledge. Obviously, this only works well for objects or areas that follow a certain color scheme. For pedestrians, this approach is not expected to produce significant results since the clothes can be any color. But for other decision factors in traffic like road lanes (here one could check for the bright/dark ratio of the street), it might help to assess whether a lane change is feasible or not due to a dashed or solid line.

Another interesting idea that not only enables some sort of conformity checks but also fosters explainability of a neural network is the combination of knowledge graphs and object detection networks. The basic process is to first detect traffic related objects in an image of the ego vehicle, then use the predicted classes of the objects to find connected objects in the graph and finally use them to explain the current traffic situation. This can help to assess the model decision and additionally provides some reasoning to why a model output might not be conform. But since this requires the integration of knowledge graphs, which is expected to cause high efforts, we decided to prioritize other approaches.

In summary, all of the methods described in the previous sections have the potential to be used for the following objectives:

- to identify cases where the conformity does not hold (flags, warnings, ...)
- to improve the performance of the AI model, by including the metric into the model.
- to improve the performance of the AI model by a more optimal train/validation split of the sample inputs

In the end, multiple of the described approaches should be combined to ensure a well-founded conformity check with regard to the training data. The approaches presented are rather basic on its own, but by combining them, they will cover multiple sub-areas (e.g. location, color, internal representations) and thus increase the significance and reliability of the resulting conformity check.

4.2 AVL Software and Functions GmbH

Topic overview:

Use Cases (Scenario)
UC1

Knowledge Source
Equations of Motion, HD Maps

Knowledge Formalization & Representation
No involvement

Conformity Check
Pedestrian Motion Model & Physical Constraints, Anomaly Detection

4.2.1 Introduction

Knowledge conformity, here, refers to the inclusion of the plausibility measures explicitly from the rule-based knowledge in data-driven models such that their training process is more efficient and the resulting predictive power is more accurate in real-world/safety-critical applications. Data-driven Artificial Intelligence (AI) models face challenges to conform to real-world scenarios which are critical in Autonomous Driving applications.

AVL proposes the development of the methods that detect the decisions of the AI that are non-conforming to formalized knowledge. These developed methods validate the AI models based on prior knowledge so that they can be re-trained in a targeted manner.

4.2.2 Use Cases

AVL focuses on the knowledge conformity for "Pedestrian Detection and Tracking" in the urban driving scenarios with various amounts of occlusions of the pedestrians as per the use-case (UC) 1 which is further split into 5 sub-cases. It considers the occlusion of the pedestrians ranging from the scenarios such as standing vehicles on pedestrian crossings, perpendicular or parallel or angular parking, intersections for right turns and parking spots for the purpose of detection/tracking.

4.2.3 Knowledge sources

Knowledge sources, for pedestrian detection and tracking, would include the physical knowledge i.e. already established domain(s) and the world & expert knowledge i.e. the experience of an individual or a group of individuals. As in the real scenarios, knowledge sources cannot be explicitly classified from one another as it depends on the interpretation of the scene therefore, we incorporate the extracted knowledge in the physical and world & expert knowledge.

We plan to use the knowledge of the position of the Bounding Box (BB) of pedestrians, and the velocity and the acceleration of both pedestrians and the ego-vehicle as the physical knowledge source. From the context of the world & expert knowledge, we plan to use the information from High-Definition (HD) maps to better understand the static environment of the ego-vehicle. In addition, we would use the publicly available datasets for autonomous driving research collected by both camera and LiDAR. These datasets should be sequential with one or more pedestrians per frame in different scenarios such as crossings, day/night timings, cities, weather, etc. It should be annotated with 2D and 3D BB for position, orientation and depth information.

Over and above that, we keep the semantic segmentation of the static environment around the ego-vehicle and lane markings or line information as an optional knowledge source.

4.2.4 Representation/Formalization concept

As per the UC 1, we plan the modelling of the Knowledge Conformity in three broad areas i.e. Physical Knowledge, World & Expert Knowledge and Combination of Methods.

Physical Knowledge: We plan to integrate the knowledge of the motion of the pedestrians in the traffic using a physical model called Pedestrian Motion Model [7]. This physical model is based on an iterative clustering algorithm using Dirichlet Process Gaussian Processes, to group trajectories into continuous motion patterns, and the transition points to identify the discrete transition points. The model combines the low-level trajectory patterns with high-level discrete transitions. Contrary to the data-driven models where constraints such as trajectory optimization or collision avoidance may produce unrealistic results, this physical model breaks down the pedestrian movement into continuous motion patterns and discrete transition points. In addition to the predictive modelling of the pedestrian motion, this physical model also

could be used for anomaly detection of pedestrian motion.

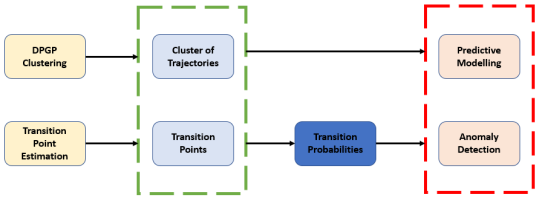


Fig. 4: Pedestrian Motion Model for Predictive Modelling and Anomaly Detection using DPGP Clustering and Transition Points.

Figure 4 [7] illustrates the block diagram of the pedestrian motion model. An iterative algorithm with Dirichlet Process Gaussian Process (DPGP) takes raw trajectory data as input for clustering and hypothesis testing to produce sub-trajectory motion patterns and estimated transition points. The sub-trajectory clusters and transition points are, then, used to find probabilities which are aimed to infer the motion patterns, transition points and transition probabilities for predictive modelling and anomaly detection.

World & **Expert Knowledge:** We shall bring in the knowledge of the static environment of the ego-vehicle using HD map/line information to understand the scenes. A better understanding of the scenes would help us to examine the expected behaviour or motion of the pedestrians and to determine the plausibility of their states and trajectory.

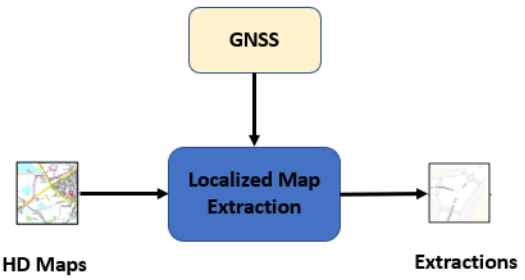


Fig. 5: Extraction from the HD maps using GNSS data.

A simple approach to conform to world & expert knowledge using HD maps can be seen in the figure 5 . The map of the city is input to the localized map extraction module. With the aid of GNSS, the localized map extraction module shall extract the section of the street where the ego vehicle is currently playing on. This extraction will have features like pedestrian crossings, side walkways and driving lanes that would be our knowledge source. Based on this knowledge, an approach to perform plausibility check can be based on anomaly detection. The detections that are output from the pedestrian detection and tracking algorithm can be fed to an autoencoder, trained on the normal pedestrian detections.

It should be able to learn the areas in which the normal detections happen, the size and dimensions of the detections (the height, width of the detected bounding box), the orientation of the detections etc. Based on these features if a pedestrian is detected whose dimensions are wider than the height of the detection, it might mean that this is an anomaly. Similarly, when we consider reflections on the windows of a building, this is an anomaly and can be detected since they are fuzzy. Moreover, the same autoencoder for anomaly detection is planned to be extended to check the plausibility of the trajectory followed by the pedestrians.

Autoencoder that shall be used to perform this anomaly detection. It shall take the input, that is the original data, and the same will be reproduced at the output in its original form. The middle layer called the core is a lower dimension representation of the actual data. The actual anomaly detection is defined, based on the magnitude of the reconstruction error that is generated. A non-normal input generates a higher reconstruction error which can be detected based on a set threshold.

Combination of Methods: Here, we plan to combine the conformity checks from both the physical knowledge and the world & expert knowledge sources. For any given scenario of pedestrian detection and tracking, two probability scores, one representing the feasibility of the scenario as per the physical laws/constraints of the ego-vehicle and the pedestrians, and another representing the feasibility with the experience of the individual(s) for a similar driving situation will be estimated. Initially, to

keep the application of the combined conformity checks relatively simpler, 50-50% weighing would be applied to both the probability scores to map two probabilities into a single score. This is planned to reduce the False Negative(s) and False Positive(s) detected and tracked by the pedestrian detection and tracking algorithm from TP1.

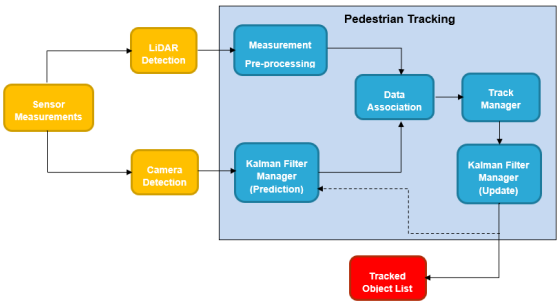


Fig. 6: Classical approach for object tracking function schema.

Object tracking, for a comparison with the algorithm from TP1, will be done using Classical Tracking Algorithm based on Kalman-filter with the following steps: data pre-processing (conversion, filter), data association, object tracking, and data post-processing. Data Association, in this scope, is measurement to track associations. Here, it associates the pre-processed measurements in the form of 3D bounding box with the objects tracked by the object tracking component. For object tracking, Kalman Filter will be used which combines the state estimations from the previous states with the current measurements. Global Nearest Neighbor (GNN) method is used for implementation of the data association function. To evaluate, precision and recall will be used. Figure 6 shows the block diagram for the proposed approach for object tracking.

The pedestrian detection and tracking network, tested for conformity check, will be integrated to the AVL's Autonomous Driving (AD) Stack for an evaluation. The integration of the perception, here, requires us to understand the inputs and outputs that are needed for the perception. The pedestrian detection and tracking algorithm takes inputs from the sensor module consisting of camera sensor models and LiDAR sensor models. Additional inputs can be received from the environment model related to the road topology. The environment used in this regard is CARLA Simulator.

The pedestrian detection and tracking algorithm shall output the detections to the AD Stack which will further evaluate the possibilities of a collision/crash with a pedestrian and take corrective measures accordingly. The outputs of the perception algorithm shall be the inputs to AD Stack:

- x, y positions
- Type
- Identity (ID)
- Motion (in x and y directions)

Here, Type refers to the class of the object that needs to be one of the inputs, and here this is class "pedestrian". This enables it to differentiate between different object classes live cars, cyclists, truck, traffic lights etc. The ID component shall enable it to distinguish between different pedestrians that are detected across the different time frames. Finally, the speed at which the pedestrian is found to be moving in x and y directions will be input to the AD Stack.

4.2.5 Conformity check

It checks how well the developed AI model complies with the different knowledge sources. It is aimed to reduce the inconsistencies and infeasibilities of the data-driven AI models to enhance the reliability and performance for pedestrian detection and tracking.

From the point of physical knowledge, the aspect-ratio and the geometry of bounding boxes over all ages, gender and parts of the globe would be considered. Moreover, the upper and the lower limits of velocity and acceleration of the detected pedestrians and the ego-vehicle would be considered. The evaluation of the AI model for conformity to the physical knowledge will be measured on GroundTruth based on Precision-Recall curve.

To conform to the world and expert knowledge, the plausibility of the localization of the detected bounding boxes would be considered. Using the information from the HD maps, inconsistent detections in the real situations such as the reflections of a pedestrian on car's windshield/windows or

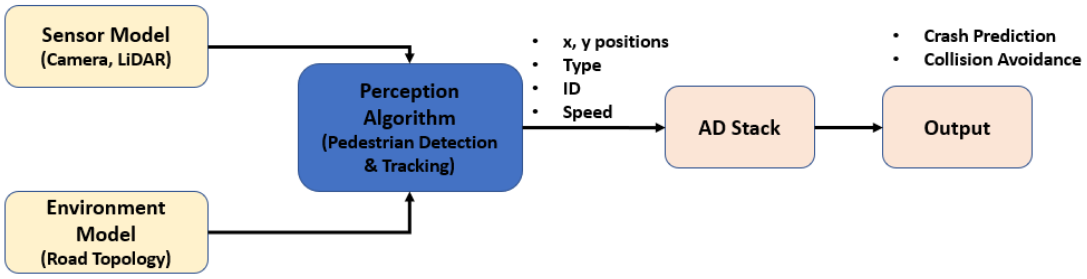


Fig. 7: Integration of Perception & Tracking Algorithm in Automated Driving (AD) Stack .

buildings (made up of glass or other shiny material) or a water puddle on the lane or the footpath need to be realized. Moreover, the improbable detections such as pedestrian detections inside a building, at a considerable height above the ground (mid-way in the air) or on the advertising board, walls, mannequins and human-shape cut-outs need to be ruled out.

Using these conformity checks, the feedback would be used for further targeted re-training of the AI models to improve the safety-critical decision making for UC 1.

4.3 Bundesanstalt für Straßenwesen (BASt)

Topic overview:

Use Cases (Scenario)
UC1, UC2, UC3

Knowledge Source
UNECE documents (ToR GRVA, ToR FRAV), RMS-1 from the FGSV, StVO, GIDAS database

Knowledge Formalization & Representation
Ontologies, Correlation analysis

Conformity Check
No involvement

4.3.1 Introduction

In this section the Federal Highway Research Institute (BASt) compiles the knowledge sources for legal and functional requirements for automated vehicles (AVs) on an international level. Furthermore, documents from the Road and Transportation Research Association (FGSV) are used as knowledge sources as these can be used to enhance and speed up the situational and scene awareness of AI-Systems by teaching standardized lane markings. Aspects of the standardized lane markings are taken and first concepts of the formalization are implemented into an ontology. In addition the design of the traffic itself is presented as a certain occlusion of road signs do not impede the validity of meaning, which can be attributed to the principle of visibility (Sichtbarkeitsgrundsatz). Another knowledge source is the German In-Depth Accident Studies (GIDAS) database [8, 9, 10], where accidents in two regions in Germany are documented in-depth with thousands of variables. A correlation analysis will give some indication, if certain variables are dependent on each other.

4.3.2 Use Cases

In this stage of the work, no specific Use Case is considered, because the legal and functional requirements are relevant for the whole system and the information gathered from lane markings can be used for perception, planning, and situational awareness.

4.3.3 Knowledge sources

Four different areas of knowledge sources are considered, which belong to the category of world and expert knowledge. The first source of knowledge are the legal and functional requirements for AVs extracted and consolidated from UNECE documents. Different working groups are gathering and formulating documents on how automated vehicles and systems should be verified and validated and how they shall behave and react on the road. Here, the most relevant documents are the Terms of Reference of the reporting group GRVA [11] and the Terms of Reference of the informal working group FRAV [12]. Of lesser importance is the NATM document [13] from the informal working group VMAD, because this document defines new assessment and testing methodologies of automated vehicles and systems and less the requirements. The requirements formulated in the Terms of Reference of the GRVA are formulated superficially. It addresses key safety aspects like system safety, object event detection and response (OEDR), safety vision, and more. The FRAV-02-05 document formulates the requirements in more detail e. g. that the system shall be able to detect the roadway, to identify lane location (w/, w/o markings), and to detect and identify lane markings. Hence, the identification of the lane markings is important to know where the roadway is.

Knowing shapes and dimensions of lane markings is necessary for their identification and these can be extracted from the guidelines for lane markings part 1 (RMS-1) and part A: Highways (RMS part A). [14] These documents were created by the Road and Transportation Research Association (FGSV) for standardized forms, shapes, dimensions, and occurrences of lane markings on German roadways. The advantage to use the RMS-1 and RMS part A documents is that one cannot only extract dimensions and shape, but also the local area (straight street, crossing, in town, out of town, and highway) in which the lane markings appear. Knowing these information beforehand and integrating them in a AI-System allows for an easier and faster perception, planning, and situational awareness. Furthermore, it creates an instance of redundancy, if the system is able to come to a conclusion with information from different sources. However, it has to be considered that streets that are older than the guidelines are not subject to these guidelines, thus not all streets follow the recommendations from these documents.

Furthermore, these documents are only guidelines and not all constructors will follow these. Meaning that it is not recommended to count solely on these guidelines.
Another source of knowledge is the German administrative regulation for road traffic regulations (VwV-StVO). [15] There, a specific source of knowledge is extracted with the principle of visibility (Sichtbarkeitsgrundsatz) which can be related to the aspect of perception and situational awareness concerning road signs. This principle says that road signs are only valid and legislative, if an average driver is able to see and perceive the road sign with a quick glance. If a road sign is occluded and its meaning is not perceivable, the road sign is not valid and legislative. However, some road signs with a certain occlusion are a special case for unfolding effectiveness. A snow-covered stop sign is still recognizable and thus unfolds its validity. The specific six-fold shape of a stop sign is only used for this sign.
Lastly, the BASt will extract relevant knowledge from the German In-Depth Accident Studies (GIDAS) database. This database contains a large set of variables that are more in-depth than regular variables, which are recorded by the police, allowing for many different analyses of accidents. The approach is to make hypotheses for possible situations that appear with set conditions and are critical and dangerous. Confirming the hypotheses with a correlation analysis shows correlations and causations between accident types and context factors.

4.3.4 Representation/Formalization concept

The knowledge extracted in the aforementioned section has to be integrated into the AI-systems, for this, it has to be represented or formalized. The representation for the lane markings will be done in an ontology. [16] This ontology shall include relationships between certain types of lane markings, their shapes, forms, dimensions, and their area of occurrence. Representing this knowledge as an ontology makes it machine readable and it is possible to integrate it into different ontologies [17] e. g. combine the ontology for lane markings with one for road signs.
Information extracted from the GIDAS database will be analyzed using the mutual information method [18] to obtain the degree of correlation between two variables. This will allow to determine, if two variables correlate strongly or weakly with each other or if they are totally independent from each other. The hypothesis based approach allows to exclude confounders and see, if there is a causal connection between two variables.

4.4 BTC Embedded Systems AG

Topic overview:

Use Cases (Scenario)
UC1, UC 2

Knowledge Source
Driving / Vehicle dynamics, Properties of the traffic Environment

Knowledge Formalization & Representation
BTC-ES' Formal Traffic Scenarios specification language

Conformity Check
Online monitoring of AI by using observers/monitors

4.4.1 Introduction

To formalize physical and mathematical knowledge of traffic dynamics, we use the scenario specification language *Formal Traffic Scenarios* (FTS), which has already been developed experimentally at BTC-ES and is constantly being extended to meet the needs of the AI Knowledge project and the use cases. FTS can be used to model, in an abstract and declarative way, a whole set of traffic scenarios that represent the use cases of the project. Based on these formalized scenarios, concrete traffic flows will be generated that satisfy the abstract scenario description. For this purpose, we use constraint solving technologies developed at BTC-ES. The generated concrete trajectories of the traffic participants in the scenario will be suitably processed in the further course of the project and integrated into the learning process as artificial training data. The formalized physical and mathematical knowledge of traffic scenarios will also be used to automatically generate monitors that will be able to monitor AI in real-time for the purpose of conformity checking.

4.4.2 Use Cases

We will mainly focus on Use Case 1, Sub-Use Case 1.2, Use-Case 2 and Sub-Use Case 2.4.

4.4.3 Knowledge sources

As a knowledge source, we will use the general description of a scenario (lanes, roads, vehicles with their phys. properties and their phys. relations to each other) and the information on the evolution of a traffic situation.

4.4.4 Representation/Formalization concept

The formalism of Formal Traffic Scenarios (FTS) [19] is a formal language for specifying traffic scenarios. FTS follows a declarative approach. This means that FTS specifies what happens in a scenario. But in terms of very concrete trajectories of vehicles FTS does not specify how it happens. In addition, a graphical representation of FTS, called *Graphical Traffic Scenarios* (GTS), has been developed by BTC-ES. In the following, GTS is used to better illustrate the concepts of FTS.

The FTS consists of several elements. These are used to formalize the traffic environment as well as to formalize the physical dynamics of the traffic participants. *The Vehicle* element describes a traffic participant involved in a scenario and in a sequence of *Traffic Phases*. The Vehicle element has a *Vehicle Type* that specifies the properties of the vehicle such as size, weight, minimum and maximum speed and acceleration. Vehicle Type is used to describe abstract classes of vehicles (car, bus and motorcycle) as well as very concrete types of vehicles or models. Traffic Phases allow to specify the dynamics of traffic participants over time. Each phase characterizes the development of a traffic situation within a limited period of time. The sequence of successive phases defines a complete scenario. Each phase includes a layout of the road (in particular the number and width of lanes) and a set of road users and their maneuvers and physical relationships to each other. Within each phase, a definition of the initial state, invariants and final state is given in terms of parameter intervals for the maneuvers and relations.

The physical and mathematical knowledge in the context of the project can be formalized in FTS by means of the so-called constraints. Currently, four different constraint types are defined in FTS. These are called *Lane Constraint*, *Distance Constraint*, *Speed Constraint* and *Speed-Difference Constraint*. The mentioned constraints in turn consist of four constraint components:

- Initial Constraint
- Invariant Constraint
- Final Constraint
- Change Rate Constraint

The idea behind these four components is il-

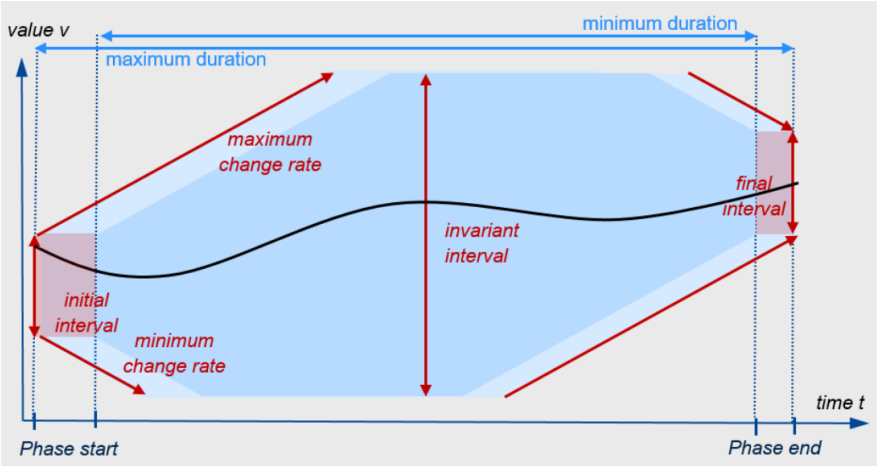


Fig. 8: Constraints concept.

lustrated in Figure 8. The x-axis represents time and the y-axis represents the value of a constraint. The black curve represents the evolution of the value of a constraint over time. The vertical red arrows at the left edge, center, and right edge of the blue area represent the allowable range of values for the initial, invariant, and final constraint at the beginning, during, and the end of a phase. The red unidirectional arrows denote the minimum/maximum rate of change of the value throughout the phase. The dotted blue vertical lines mark the start range and the end range of the considered phase. The phase can start as soon as the value is contained in the specified start interval and it can end as soon as the value is contained in the start interval and the actual duration of the phase is between the allowed minimum and maximum durations (represented by the horizontal light blue arrows at the top of the figure). The light blue area marks the allowable values at specific times considering all intervals and rates of change, i.e. if the black curve would leave this area, a constraint is violated. However, the rate of change also applies to values within this range, i.e. if the slope of the black curve comes out of the minimum/maximum rate of change, a constraint is also violated.

The *Lane Constraint* refers to the position of a vehicle on a lane within a certain phase. The two most common variants of the constraint are the *Lane-Keep Constraint* and the *Lane-Change Constraint*. The former specifies that the vehicle remains in the same lane throughout the phase. This is equivalent to specifying only an invariant constraint for the lane position of a vehicle and not an initial or final constraint. On the other hand, the second variant is used to specify that the vehicle performs a lane change within the phase. In this case, in addition to the invariant lane constraint, the initial and final lane must also be specified.

The *Distance Constraint* refers to the longitudinal distance between two vehicles on a lane. As with the lane constraint, it is again possible to specify the initial, invariant and final constraints as well as the speed at which the distance may change over time.

Speed Constraint refers to the absolute speed of individual vehicles and can in turn be specified as an initial, invariant and final constraint as well as a rate of change corresponding to the permissible acceleration of the vehicle.

Speed-Difference Constraint refers to the relative speed difference between two vehicles and can again be specified as Initial, Invariant and Final Constraint as well change rate of the speed difference.

The formalization of the physical constraints of the traffic participants in a scenario formalized with FTS results in temporal formulas containing

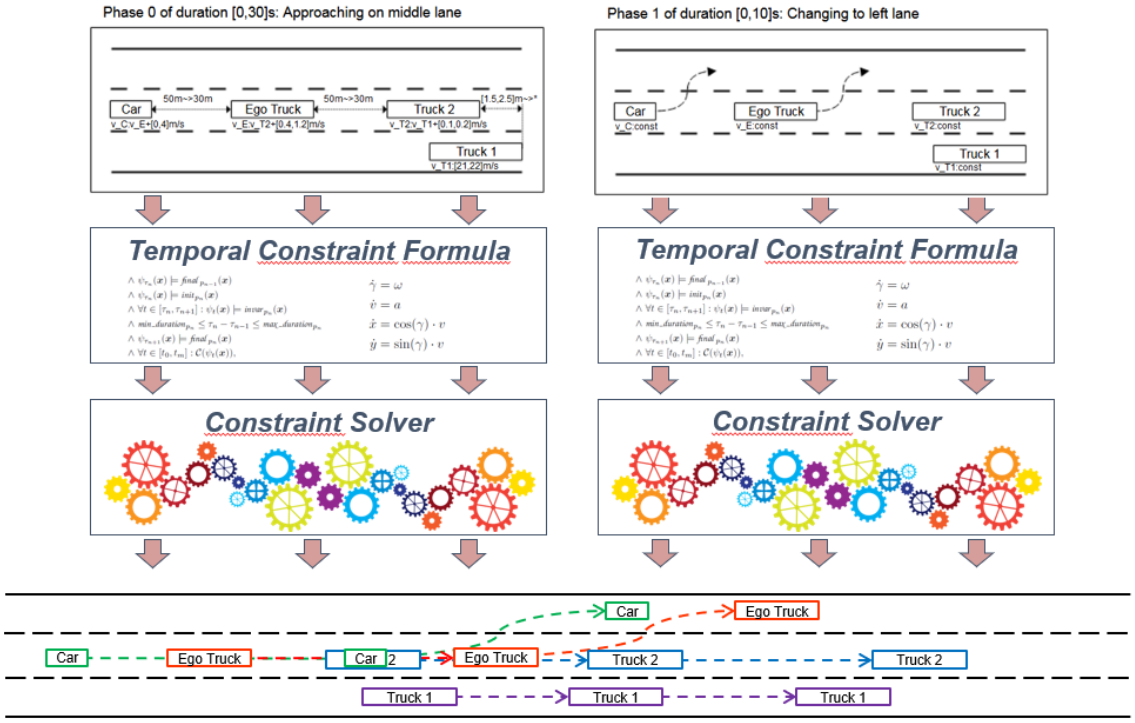


Fig. 9: Generation of possible trajectories using constraint-solving technologies.

differential equations. These equations are derived from the driving dynamics of the traffic participants. The mentioned constraint solving technologies [19] are able to solve these equations automatically. The result is available in the form of possible values of the physical quantities of the traffic participants (position, speed, etc.) that satisfy these temporal formulas. Based on this information, a possible trajectory can be created for each traffic participant in the scenario, see Figure 9.

In addition, our concept foresees the implementation of a toolbox based on FTS and constraint-solving technologies for the generation of concrete traffic flows in the form of possible trajectories of traffic scenario, cf. Figure 10. The toolbox formalizes, by means of FTS, the information underlying a scenario about the available vehicles, the traffic environment, and the physical constraints. In the next step, the toolbox automatically derives a set of possible trajectories from the formalized scenario using constraint solving technologies.

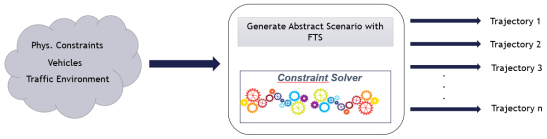


Fig. 10: A toolbox for generation of trajectories.

It is planned to process the generated trajectories appropriately and to make them available as artificial training data for the AI models.

Furthermore, the concept foresees the integration of distribution functions into the toolbox. A distribution function for a realistic distribution of characteristics of lane changes from UC2 makes it possible to express that, for example, abrupt lane changes are rather rare.

By integrating proper distribution functions,

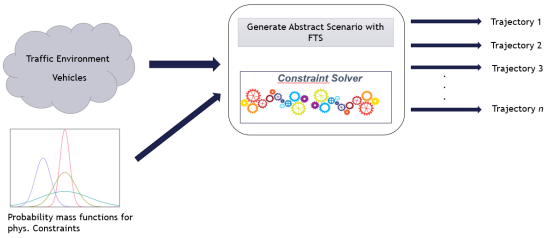


Fig. 11: Integration of the distribution functions of the phys. Constraints into the toolbox.

it would be possible to generate any number of manifestations of a particular desired traffic flow. One approach to setting up the distribution functions is to use existing data sets as a basis. Our assumption is that the physical constraints that can be extracted from a data set correspond to a certain distribution.

4.4.5 Conformity check

Our approaches focus on generating monitors from formalized physical and mathematical knowledge of traffic scenarios. The generated monitors are able to monitor AI online. Occurring violations of physical laws or other inconsistencies can thus be detected in real time, especially violations of the allowed driving behavior of the ego-vehicle in exceptional situations. This information can be used for offline re-training of the AI as well as for influencing the AI in real-time. The formalization of the phys. and math. knowledge of traffic scenarios is performed using BTC-ES' scenario formalism. The formalism has already been discussed in detail in the paragraph *Representation/Formalization concept*. The story-line for the concept of conformity checks is shown in Figure 13.

The concept for conformity checking essentially consists of three elements:

- Formalized physical and mathematical knowledge
- Monitor interface
- Monitor

In order to implement the basic idea described in the introduction, the first step is to use the BTC-ES scenario formalism to formalize a traffic scenario for which a monitor is to be generated. The formalized scenario is given in the form of a FTS. The FTS contains a formalized representation of the traffic environment (vehicles, road, lanes, ...) of the scenario as well as the physical dynamics of the traffic participants. In particular, it formalizes the maneuvers of the individual traffic participants, their physical relationships to each other (e.g. distance and speed differences), and the physical characteristics of the participants (e.g., acceleration and speed).

Figure 12 depicts a graphical representation of the FTS from Use Case 2. 4. Here, the following physical knowledge is formalized:

- The ego-vehicle (blue) changes lanes (Lane Constraint)
- The traffic participants (gray) maintain certain velocities (Speed Constraint)
- The ego-vehicle must maintain a distance of 5 - 10 m to the vehicle in front (Distance Constraint) before changing lanes

Our concept is that the monitor interface receives the formalized traffic scenario in the form of an FTS and identifies the maneuvers occurring in the scenario as well as the physical characteristics of the traffic participants and their relationships to each other. In the next step, the monitor interface derives a set of rules for all constraints identified in the FTS. Referring to the distance constraint in Figure 12, a derived rule may be as follows: "The ego vehicle must always maintain a distance between 5 m and 10 m to the vehicle in front."

Following this, the monitor interface generates the actual monitor to be used for online monitoring of the AI. The monitor will be available in the form of code (Java, C, etc.). This monitor will be connected to the demonstrator, in which a traffic scenario is simulated. The monitor should receive so-called stimuli data from the demonstrator in real time. This data contains phys. information of the traffic participants in the simulation, such as the current speed, acceleration, position, etc..

The task of the monitor is to ensure that the constraints formalized in the FTS are satisfied. In doing so, the monitor uses the stimuli data received from the demonstrator to check that the traffic participants in the simulation do not violate the rules derived from the FTS. If there is a violation

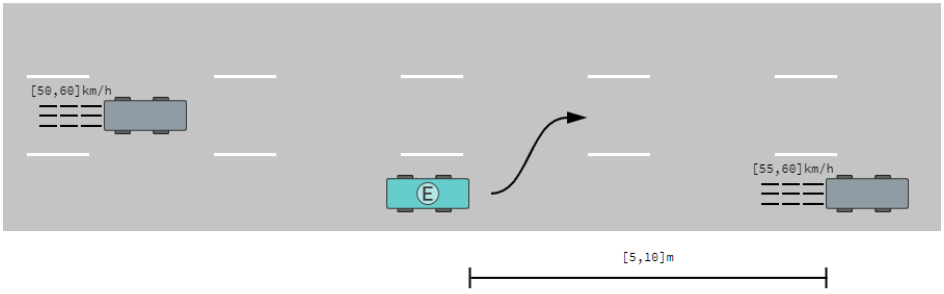


Fig. 12: Graphical representation of a distance constraint between the ego-vehicle and another vehicle in front in UC 2.4.

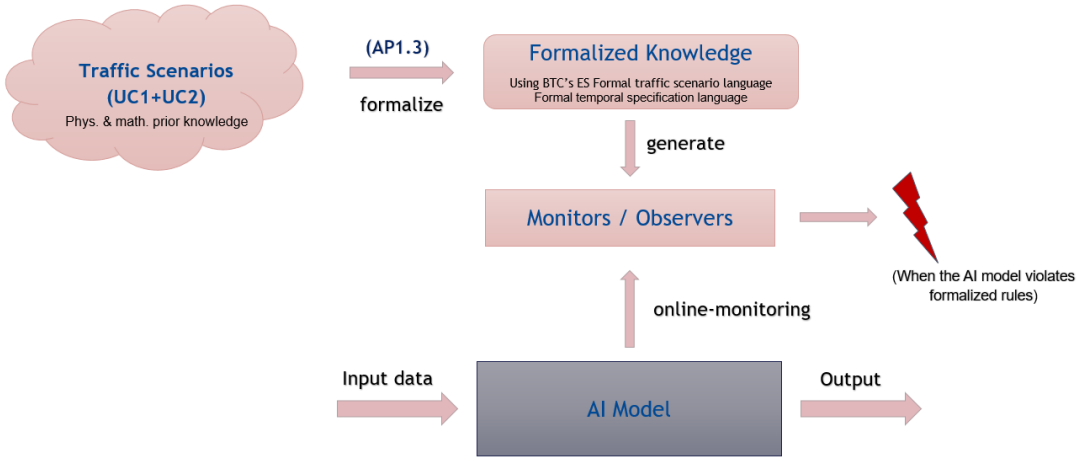


Fig. 13: Concept of conformity checking.

of a constraint, the monitor sends feedback to the demonstrator about the type and reason for constraint violation. Based on this information, an AI can be influenced in real time to re-plan the trajectories so that the constrains are satisfied again. The described concept of monitor generation from an FTS is depicted in Figure 14.

The pseudo-code in Figure 15 represents an example of a very simple monitor that could be generated from the FTS in Figure 12. The monitor receives information from the demonstrator about the current distance between the ego vehicle and the vehicle in front on the same lane. The monitor checks if the distance constraint is satisfied.

In addition, our concept offers the possibility to specify a tolerance range for the monitor in which the constraints are allowed to be violated.

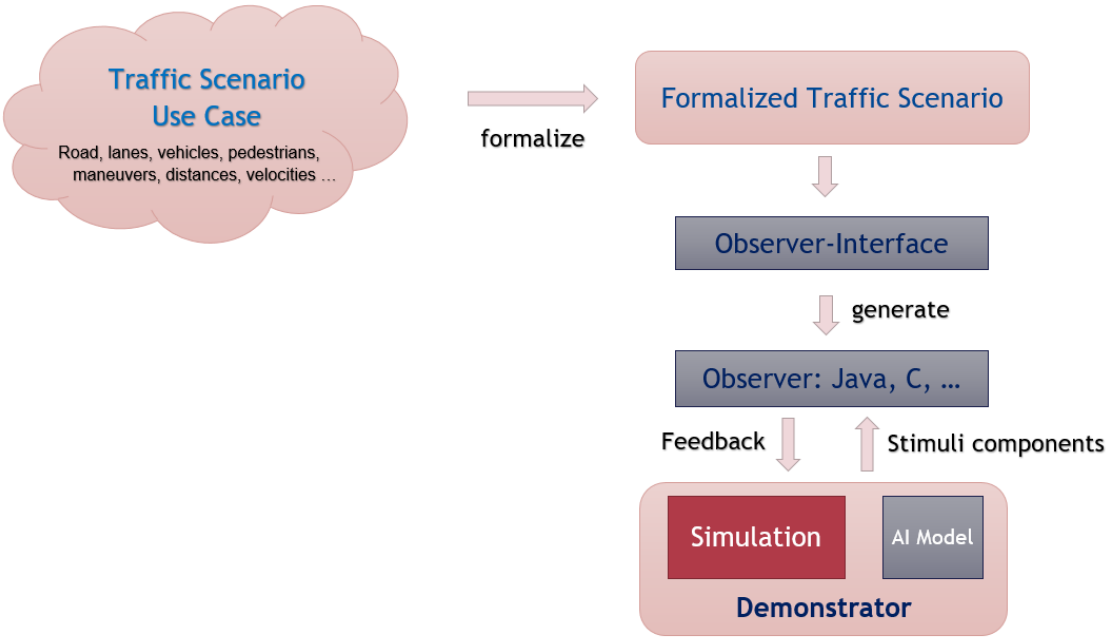


Fig. 14: Concept of monitor generation from an FTS.

```
While(ego_is_driving):
      x = getDistanceToVehicleInFront( ); // Distance between ego vehicle
      and the vehicle in front
      if x ∈ [5, 10]:   // Check if distance is within the constraint
                        interval
      // Keep Driving
      else:
          raiseConstraintViolation( );  // A constraint is violated
          sendFeedbackToAI( );  // Feedbacl to AI about the violation
```

Fig. 15: Pseudocode of a very simple monitor derived from FTS.

4.5 Capgemini Engineering

Topic overview:

Use Cases (Scenario)
UC1, UC 2

Knowledge Source
semantic map of static environment

Knowledge Formalization & Representation
image of static semantic map

Conformity Check
Verifying semantic detection regions, physical feasibility check

4.5.1 Introduction

In urban environments, the layout of streets and traffic rules play an important role in predicting the presence of pedestrians (Proposal 1 for Use Case 1) and in predicting the trajectories of traffic participants (Proposal 2 for Use Case 2). In this work we introduce combined solutions that leverage information from both the raw input data and the environment for the two use cases.

For proposal 1, we introduce deep-learning approaches that yield from state-of-the-art neural network architectures for object detection and 2d semantic segmentation. The Mask R-CNN [20] is well-known for instance segmentation problems. The network unfolds to a combination of two sub-architectures; the Fast R-CNN [21] and Region Proposal Network (RPN) [22] contribute to an efficient feature extraction from the input data. The Region of Interest (RoI) Alignment outperforms the conventional RoI pooling in pixel-to-pixel accuracy since there's no quantized striding. For each RoI, a fully convolution network is applied for segmentation mask prediction, in parallel to the existing classification and bounding box regression branch.

For proposal 2, a neural network is designed to predict the trajectories of traffic participants. The network is a combination of a Long Short-Term Memory (LSTM) and a Convolutional Neural Network (CNN). The input data for the LSTM part of the network are the past trajectories of the car, whose trajectory will be predicted, and the surrounding traffic participants. This LSTM encoder extracts relevant features from the past trajectories. The input of the CNN part of the network is an image of the street environment and extracts features of the local street layout. The extracted features of the past trajectories and the street image should support the network to make predictions according to the street layout.

4.5.2 Use Cases

As we are mainly using road maps and semantic images as prior knowledge in our two conceptual proposals, we will be able to cover different scenarios under both Use Case 1 (Pedestrian Detection) and Use Case 2 (Lane Change) using the CARLA simulator [23].

Use Case 1: Improved pedestrian detection under occlusion can help drivers or ADAS (Advanced Driver Assistance Systems) to locate pedestrians and timely react to ensure the traffic safety. In this project and as a start, the focus will be on the traffic situation in urban space and namely on Use Case 1. We will start with a simple scene that consists of a road of two straight lanes without oncoming traffic, a parking lane with some standing cars and a pavement where few pedestrians walk. We then want to increase the complexity of the scene to handle more sophisticated concrete scenarios. In practical application, occlusion is common in crowded streets where the movement of pedestrians and the change of environment bring great challenges to the detection algorithm. The following aspects would be considered to include variations on scenarios:

- Types of occlusion objects (cars, vans, bushes, dumpsters, advertising poster, etc.), their number, sizes, and poses
- Type of pedestrians (adults, children, wheelchair users, etc.), their number, speeds and movement
- Change in weather conditions
- Speed of the AD-Vehicle (Automated Driving Vehicle)
- Surrounding environment including road type, e.g. lane width, straight or curved road, etc.

Figure 16 shows our concept on how to integrate our developed KI-Module (KI stands for Künstliche Intelligenz which is German for Artificial Intelligence) in the architecture of the Demonstrator for Use Case 1. We use CARLA as a simulation environment and ROS (Robot Operating System)

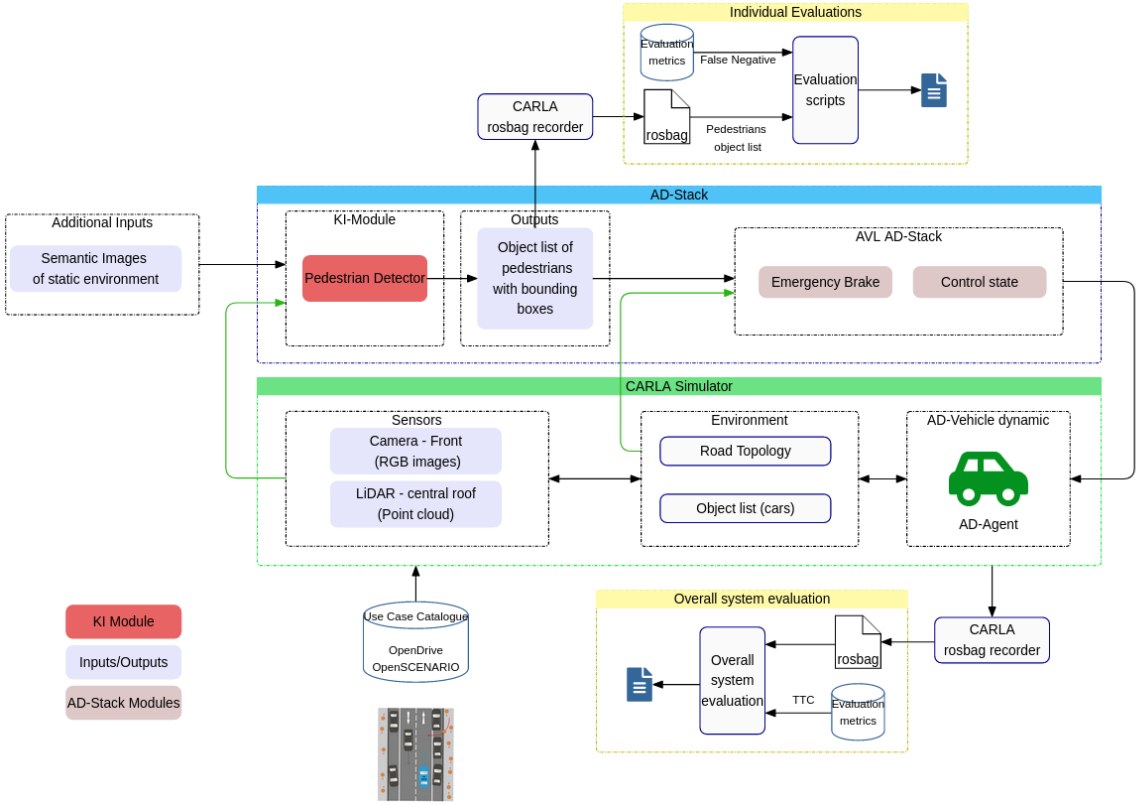


Fig. 16: Demonstrator architecture for Use Case 1

for communication with the different modules. The KI-Module (Pedestrian Detector), will receive (subscribe) RGB and semantic images and output (publish) an object list of segmented pedestrians with their bounding boxes. The semantic images will be prepared separately in a prior step. The object list of pedestrians together with road topology from simulation will be used then by the AD-Stack to activate the emergency brake and control the AD-Vehicle to closed the simulation loop. In order to evaluate the results achieved within the scope of this Use Case, it is therefore essential to evaluate the KI-Module both individually and in the context of an overall system using the suitable evaluation metrics.

Here are the main parts of the Demonstrator which we need for the integration:

- The CARLA simulation environment
- The concrete scenario described in OpenSCENARIO
- A scene map taken from one of the default towns available in CARLA
- The KI-Module (Pedestrian Detector)
- The individual evaluation metrics which evaluate the performance of the KI-Module, for example to calculate the false negatives (FN)
- The overall system evaluation metrics which evaluate the performance of the KI-Module in the whole scenario, for example to calculate the time to collision (TTC)
- The AD-Stack i.e., the additional functions which you may need to run/test the KI-Module. In our case we will need the emergency brake and the control functions

Use Case 2: Trajectory prediction of surrounding traffic participants will facilitate planning of

complex lane change of the AD-Vehicle in both urban and highway scenarios. Related to this aspect, Capgemini Engineering will work on Use Case 2 (Lane Change) and focus on the prediction part without planning. As in Use Case 1, we will start with a simple scene that consists of a highway road with three straight lanes, one slow car driving in front of the AD-Vehicle and a faster car approaching on the middle lane. The following aspects would be considered to include variations on the scenario:

- Location of the AD-Vehicle on the road, i.e., on lane0, lane1 or lane2.
- Dynamics (pose, speed, acceleration) of AD-Vehicle and all other traffic participants.
- Geometric design of the road, i.e., straight or curved lanes.
- Traffic density on different lanes.

The demonstration concept for Use Case 2 will be provided in the result of the next project increment.

4.5.3 Knowledge sources

Our proposals use world knowledge as knowledge source, where the world knowledge is represented as a static semantic 3d map around the ego-vehicle. The static environment information around the ego-vehicle is seen as prior knowledge in pedestrian presence estimation and trajectory prediction, as it exists independently from and yet has a great influence on the actual pedestrian presence and future trajectory.

The prior knowledge in a brief description is a 3d map surrounding the ego vehicle which contains exclusively static objects and interesting semantics. Such a map can be acquired by accumulating lidar scans over time and transforming the point-clouds into global coordinates via localization information. Moving points with respect to world are filtered out thus only static points are left for building the map. The map is on the point level labeled with semantic labels. A comparable representation of the map is shown on the right side of figure 17. Another way to get the 3d map of the ego vehicle surrounding is to query street information from an online geographic database like OpenStreetMap [24]. An approach for street-map based validation for semantic segmentation using OpenStreetMap is presented in [25].

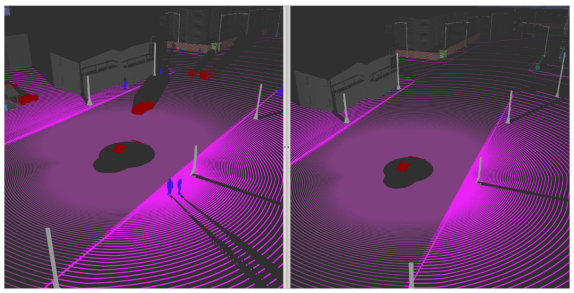


Fig. 17: Map with and without dynamic objects. Each semantic label is represented with a different color

4.5.4 Representation/Formalization concept

As mentioned in the previous chapter, both proposals use world prior knowledge in the form of a static semantic 3d map of the ego-vehicle surrounding. To integrate this knowledge into the model for pedestrian detection or trajectory prediction, the 3d map information will be projected into a virtual camera image.

In proposal 1, the virtual camera is mounted at the same spot as the RGB camera and has an identical camera model and parameters. Using the same camera configuration for the prior and RGB image there is a one-to-one match between both images and the model do not have to learn the physical transformation between them.

In proposal 2, the virtual camera without perspective is mounted as a bird's-eye view camera and it captures the street environment in front of the vehicle, whose trajectory is predicted. The orientation of the bird's-eye view image corresponds to the driving direction of the ego vehicle.

4.5.5 Conformity check

The conformity check aims at checking if the detected objects and predicted trajectories are consistent with prior knowledge, with the underlying goal of improving the model robustness and accuracy.

For proposal 1, the presence region of detected bounding boxes will be checked to determine if the detections appear in valid semantic regions. A bounding box floating above a building is highly probable a false positive detection, as can be seen

Fig. 18: Schematic representation of detected bounding boxes and masks (top) and corresponding prior static semantic image (bottom)

in figure 18. To realize this check, valid semantic regions are defined, and the predictions are overlayed with the prior static semantic image. In this way a score can be calculated how likely a prediction in the semantic region will be.

Another concept for conformity checks is to run the inference for pedestrian detection using the same RGB image and different prior images as input. The outcome of the network should be different when the prior image just represents an environment full of buildings compared to an environment full of sidewalks.

After describing the concepts for conformity check for proposal 1, the developed concepts for conformity check for proposal 2 are outlined. A similar check compared to proposal 1 is to verify if the predicted trajectories are in valid semantic regions (e.g. road, sidewalk vs. building). This check assumes that the cars obey the traffic rules and drive on the road. Furthermore, the prior image of the street environment can be replaced with an image that does not contain any street environment information (e.g. black image) and review how it will affect the prediction.

Furthermore, concepts are developed to check if the predicted trajectories are physically feasible. This can be realized by comparing the prediction with different motion models. Valid motion models can be defined by constraints on velocity, acceleration, and yaw rate.

4.6 Continental AG

4.6.1 Continental - World & expert knowledge

Topic overview:

Use Cases (Scenario)
UC2, UC3
Knowledge Source
Traffic Rules, Driving Simulator Data
Knowledge Formalization & Representation
Text & Knowledge Embeddings
Conformity Check
No involvement

4.6.1.1 **Introduction**:

The goal of Continental Automotive GmbH in the context of world and expert knowledge integration in to autonomous vehicles is to formalize existing knowledge via subsymbolic representations. This type of representation can be seen as n-dimensional, numerical vectors, i.e. embeddings, which serve either as input into neural network architectures or can be seen as representations from intermediate network layers. Creating embeddings based on knowledge items, such es road traffic rules, overcomes one of the existing problems in hybrid AI solutions, the representational gap. Due to the variety of ways to formalize existing knowledge, e.g. text, knowledge graphs, logical systems, downstream architectures typically need to consider and address all the specific properties and problems of all the underlying formalism. Presenting knowledge as subsymbolic representations allows downstream architectures to easily integrate knowledge into the main use case architecture, e.g. a trajectory prediction algorithm, without focusing on the knowledge integration part. A drawback compared to explicit knowledge models is the lack of interpretability of embeddings. Thus, we plan to investigate how our developed embeddings can be evaluated towards their expressiveness regarding reflecting world and expert knowledge.

Another goal of our research activities is to investigate how to resolve conflicting norms. For instance, it might be necessary to ignore a specific rule to reduce the likelihood of a dangerous traffic situation or accident. This is only possible if the current traffic scenario with all involved agents is considered during the resolving process.

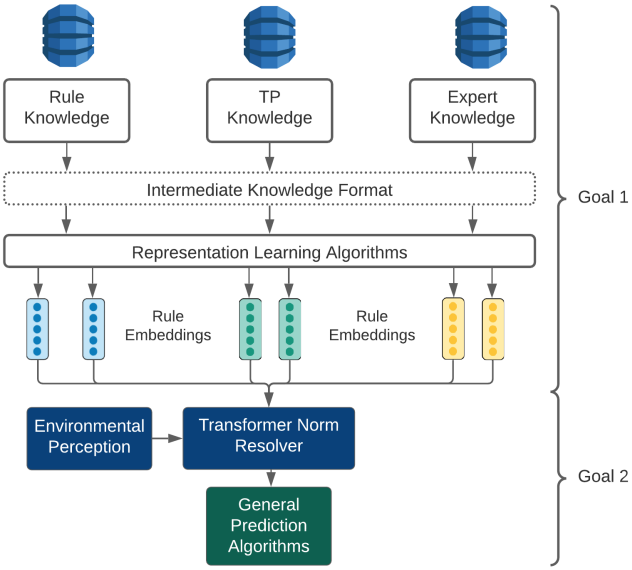


Fig. 19: An overview of Continental's approach. Given a set of initial knowledge sources and their transformation into a possible intermediate format, we develop representation learning algorithms to convert knowledge into subsymbolic representations (i.e. Goal 1). It follows a norm resolving unit to resolve conflicting norms and rules given a traffic situation. Our output is agnostic towards any downstream architecture and use case (i.e. Goal 2).

Generally, the overall goal is to provide knowledge building blocks that can be used for any follow-up architecture and use case. However, we plan to integrate and demonstrate our approach in the context of a parallel developed trajectory prediction algorithm.

Figure 19 summarizes our solution and indicates how our goals depend on each other. We don't know yet if or which kind of intermediate knowledge representation we need to come up for the best embeddings. This is sketched by the dotted line. Resolving conflicting norms can be seen as an incremental step and is addressed as soon as we developed a first set of sophisticated embeddings.

4.6.1.2 **Use Cases**:

We envision the integration of world and expert

knowledge into downstream tasks (e.g. trajectory planning) as the guiding light for our application scenarios, which support both our use cases. In particular, we focus on the following two driving scenarios:

1) Crossroads Scenario: Crossroads typically require the complex resolution of a set of traffic rules in the context of multiple traffic agents.
2) Lane Change Scenario: Successfully predicting and changing the lane on a highway with multiple traffic participants can only be handled by an ego vehicle if traffic rules are carefully considered along the planned trajectories.

In both scenarios, the Crossroads Scenario and the Lane Change Scenario, the traffic conditions require the ego vehicle to deal with a set of specific conditions. In the following, we distinguish between regular and irregular conditions, which are also illustrated in Figure 21:

- Regular conditions: Traffic rules play an essential part in nearly all driving scenarios. Thus, our described scenarios can be perfectly leveraged to test and evaluate the integration and consideration of normative knowledge in autonomous systems. To name a concrete example: vehicles are typically free to continue left, straight, or right at a crossroads. Disallowed actions, however, may stem from the fact that traffic signs limit the entrance into one of these directions, e.g. disallow turning right because it is a one-way street in the opposite direction. Our goal is to share this knowledge about traffic signs and traffic rules in their subsymbolic representations with the trajectory prediction algorithm.
- Irregular conditions: We will also focus on exceptional circumstances that trigger non-standard actions by autonomous vehicles. These circumstances require the correct resolution of a given set of conflicting traffic rules to prevent dangerous situations. For example, an autonomous vehicle may be forced to stop at a green traffic light (Rule 1: Drive when green) when an ambulance vehicle with blue lights and sirens on (Rule 2: Carefully provide road space and let ambulance pass), is crossing its way.

With these information, the assignment of our scenarios towards use cases is fairly simple. While our Lane Change scenario covers Use Case 2 and 3, the Crossroad scenario is connected with Use Case 3 only.

4.6.1.3 **Representation/Formalization concept**:

Goal 1: Transformation of normative knowledge into subsymbolic representations. The first part of our contribution addresses the transformation of normative knowledge (i.e. traffic rules) into subsymbolic representations (i.e. embedding vectors). For normative knowledge sources, we will initially target descriptions of traffic signs as simplified instances of traffic rules, whereby each description represents a short text snippet stating the purpose of the sign. In the course of development, we plan to scale our efforts to more elaborate sets of rules such as the German *Straßenverkehrsordnung (StVO)*.

Transforming textual knowledge into subsymbolic representations has been widely studied in the natural language processing literature. Recent approaches based on the Transformer [26] such as BERT [27], RoBERTa [28], DistilBERT [29], ALBERT [30], or XLNet [31] demonstrate the potential of leveraging large pre-trained language models for language understanding tasks. Using these models, we are able to encode world and expert knowledge in subsymbolic representations—with the resulting embedding vectors meant to be expressive representations of the semantic contents.

We have already conducted initial pilot studies to this end. Given a set of rules for road signs, we leveraged pre-trained language models to create embeddings directly from text without bringing the text into intermediate representations. Our evaluation regarding the embedding similarity in the vector space showed two major challenges: (i) Embeddings for traffic rules are typically in a similar location in the vector space although they represent completely different rules. This can be explained with traffic rules belonging to the same domain, e.g. automotive. The underlying model was trained on a wide variety of domains, however. (ii) Traffic rules formulated as law text are quite general in their formulation which results in non-concrete representations.

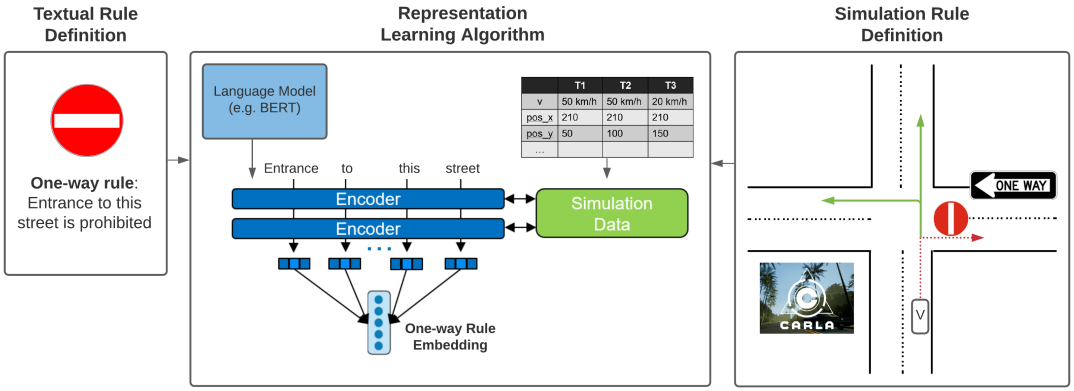


Fig. 20: General architecture of Continental's approach to combine traffic simulation data with language information to generate embeddings. We pre-proccess the initial textual representation of the *One-way rule* with natural language processing tools (cf. left rectangle). Then, we extract rule information from a simulation environment like Carla [23] which allows us to reflect the rule within a traffic scenario (cf. right rectangle). The developed representation learning algorithms allow us to combine the different knowledge types in an algorithm to create semantic embeddings.

We conclude that these findings indicate that we will need to go beyond encoding solely textual contents as the resulting embedding vectors of formalized traffic rules appear to fall short of being sufficiently expressive for the use cases at hand. For that reason, we plan to leverage encoded world and expert knowledge alongside additional data, for which we mainly consider structured data gathered from user studies in driving simulators. Figure 20 shows a conceptual example how we are planning to encode the traffic sign *One-way rule*. First, we take and pre-proccess the initial textual representation of the rule with common natural language processing tools (cf. left rectangle). In a second step, we extract relevant rule information from a simulation environment like Carla [23] which allows us to reflect the rule within a specific traffic scenario. In the right rectangle, we chose the one-way example on a crossroads. For each time period T, we select and extract a set of traffic environment information like the velocity v or the ego vehicle position (pos_x, pos_y). The relevant information that are capable of improving the embeddings are yet to be found. However, we see two possible ways to integrate simulation data into the embeddings creation process: i) Affecting the training process of embeddings with simulation data, for instance, by including the information in the loss function, and ii) adapting the model architecture in such a way that simulation data is directly integrated into pre-trained language models. There is substantial recent work on integrating factual knowledge into pre-trained language models, e.g. KEPLER [32], K-Adapter [33], CoLAKE [34], or [35]. To the best of our knowledge, at present the main body of research focuses on well-structured knowledge sources such as knowledge graphs, syntax trees, or named entity information for the integration with large pre-trained language models, and the overarching aim is to enrich these language models with external information.

It is an open research question which way of integrating external information with language models allows us to best solve the use cases at hand, and we will investigate this question as part of our contribution.

Goal 2. Resolution of conflicting norms in exceptional situations. The second part of our contribution focuses on the resolution of conflicting norms (i.e. traffic rules) in case the current road conditions

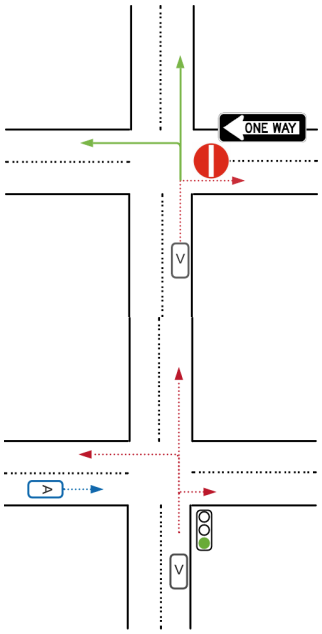


Fig. 21: Illustration of the Crossroads Scenario for regular (left) and irregular conditions (right). Disallowed actions may stem from the fact that traffic signs limit the entrance of vehicle V into one direction (regular conditions). Exceptional circumstances such as a passing ambulance vehicle A require the correct resolution of a given set of conflicting traffic rules to prevent dangerous situations (irregular conditions).

require it (e.g. in case of an accident, a passing ambulance vehicle, or similar).

In both the Lane Change as well as the Crossroads scenario, we assume that autonomous vehicles will need to take decisions from time to time that involve intentionally breaking traffic rules as a resolution of conflicting norms (cf. Paragraph 4.6.1.2). Encoded world and expert knowledge is meant to raise the situational awareness of autonomous vehicles in order to determine which type of maneuver likely yields the best outcome.

Similar to Goal 1, we envision our realization to build on a Transformer architecture with self-attention mechanisms [26].

We will conduct experiments that jointly provide encoded traffic rules as well as structured data from driving simulators as input to the Transformer. We plan to fine-tune the model for simulated driving situations that require resolving conflicting norms as outlined in Paragraph 4.6.1.2. We expect our model to learn how to weight different norms in given situations, and determine which rules to best override as the situation requires it.

Continental - UC2, UC3 - Knowledge on Scene & Evolution - Equations/Inequalities - Equivariance Constraints, Likelihoods

Topic overview:

Use Cases (Scenario)

UC2, UC3

Knowledge Source

Physical & Mathematical Knowledge

Knowledge Formalization & Representation

Parametric representation, priors over parameters

Conformity Check

Equivariance constraints, likelihoods

4.6.2 Continental - Mathematical & Physical Knowledge

4.6.2.1 **Introduction**:

With the aim of developing a scenario tracking and long therm prediction system, Continental's work in AP1.3 focused on the development of a universal trajectory representation for traffic participants. This is done in the context of state space models (SSMs) that will provide the scaffold of an algorithmic prior of Bayesian Data Assimilation [36] with modular exchangeable components for motion and observation models developed within AP1.2. The individual building blocks of SSMs, i.e. state space, observation model, motion model can then be altered individually by components that leverage data and learning in their parameterization. In this work, we focus on the state space representation and derive two equivariance constraints for the motion model.

4.6.2.2 **Use Cases**:

The trajectory representation is primarily intended to use a state representation for traffic participants within the scope of a long term traffic scene prediction system developed in AP1.2. As such, it will play a rôle in the respective use cases addressed there, i.e. predicting the trajectories of traffic participants to enable anticipatory planning in complex driving maneuvers on highways and at intersections (UC2), as well as the detection of rule violations in (UC3).

4.6.2.3 **Knowledge Sources**:

We will employ primarily physical and mathematical knowledge. The realization that agents to move smoothly will allow us to formulate compact parametric trajectory representations. We can even model the extent of smoothness in our representation by placing priors on its parameters. The fact that physical agents cannot change their state of motion arbitrarily fast will give us solid mathematical bounds on the worst case errors we can expect and the preference for comfort will limit the typical case errors. We further derive two equivariance constraints that are in place in any Bayesian filtering architecture and thus also apply to ours.

4.6.2.4 **Representation/Formalization concept**:

The key concept here is to promote object trajectories to Markov states in the framework of state space models. The state vector then becomes the parameter vector of our trajectory representation. The motivation for this is that an agent's intent is encoded in its past trajectory and instead of inferring latent intent variables, we'd like to use the past trajectory directly as input in predicting future motion. The parametric formulation gives us compactness of representation and computational efficiency.

4.6.2.5 **State Space Models**:

Driving is a sequential process. An agent receives a steady stream of observations $\mathbf{o}_t$ through its sensors. These observations are then used to plan an agent's actions forward in time. This, in turn necessitates the ability to form expectations about the future of the environment. The only way to verify such expectations is through comparison with future observations. Thus, the ability to model future observations $\mathbf{o}_{t+\delta t}$ given past observations $\mathbf{o}_{0..t}$ via $P(\mathbf{o}_{t+\delta t}|\mathbf{o}_{0..t})$ is crucial.

Many data driven systems attempt to model $P(\mathbf{o}_{t+\delta t}|\mathbf{o}_{0..t})$ through $P(\mathbf{o}_{t+\delta t}|\mathbf{o}_{t-\Delta t..t})$ with a limited look-back time Δt. In contrast, state space models introduce a nuisance or latent random variable $\mathbf{x}_t$ called *state* with the *defining* property

$$P(\mathbf{o}_{t+\delta t}|\mathbf{o}_{0..t}) = \int \mathrm{d}\mathbf{x}_{t+\delta t} P(\mathbf{o}_{t+\delta t}|\mathbf{x}_{t+\delta t}) P(\mathbf{x}_{t+\delta t}|\mathbf{o}_{0..t})$$

Though looking innocuous, this equation has wide implications. The state renders past and future observations conditionally independent. The state "shields" the future from the past. It captures *all* information necessary to predict future observations from past observations, i.e. it considers, at least in principle, a potentially infinite look-back time [37, 38]. It makes the state the sole cause of observations. The above equation is of little use, however, if we

don't have access to the so-called *predictive state density* $P_{t+\delta t|t} \equiv P(\mathbf{x}_{t+\delta t}|\mathbf{o}_{0..t})$. Thus, SSMs introduce the additional so-called Markov-constraint on the state that it shall be the sole origin of all dynamics in the system:

$$P(\mathbf{x}_{t+\delta t}|\mathbf{o}_{0..t}) = \int \mathrm{d}\mathbf{x}_t P(\mathbf{x}_{t+\delta t}|\mathbf{x}_t) P(\mathbf{x}_t|\mathbf{o}_{0..t}).$$

Here, we have introduced the *motion model* $P(\mathbf{x}_{t+\delta t}|\mathbf{x}_t)$ as the generator of all system dynamics and the *posterior state density* $P_{t|t} \equiv P(\mathbf{x}_t|\mathbf{o}_{0..t})$ to express our knowledge about the state given all observations up to and including time t. It is this quantity that can be tracked over time by alternating the prediction step of forming the predictive state density $P_{t+\delta t|t}$ (1) and the update step for the posterior state density $P_{t+\delta t|t+\delta t}$ (2) by absorbing new measurements via the application of Bayes' rule:

$$P_{t+\delta t|t} = \int \mathrm{d}\mathbf{x}_t P(\mathbf{x}_{t+\delta t}|\mathbf{x}_t) P_{t|t} \quad (1)$$

$$P_{t+\delta t|t+\delta t} = \frac{P(\mathbf{o}_{t+\delta_t}|\mathbf{x}_{t+\delta t}) P_{t+\delta t|t}}{P(\mathbf{o}_{0..t+\delta t})} \quad (2)$$

The observation likelihood $P(\mathbf{o}_{t+\delta t}|\mathbf{x}_{t+\delta})$ is also called *observation model* and the evidence term $P(\mathbf{o}_{0..t+\delta t})$ in the update equation plays the role of a normalization constant for the posterior state density. With these equations, a more refined state estimate can be obtained over time through data assimilation.

4.6.2.6 **State Density and State Density Updates through Bayesian Filtering**:
The filtering equations demand that we are able to marginalize over $\mathbf{x}_t$ in the prediction step (1). Further, it is desirable that the posterior state density $P_{t+\delta t|t+\delta t}$ maintains its functional form under a the update step.

There are two extreme ways to ensure this. The first is known as the Linear Gaussian Approach. This means that we choose a state space $\mathbf{x}_t \in \mathbb{R}^d$ and model the predictive and posterior state densities as multivariate normal distributions:

$$\begin{aligned} P_{t+\delta t|t} &= \mathcal{N}(\mathbf{x}_{t+\delta t|t}; \boldsymbol{\mu}_{t+\delta t|t}, \boldsymbol{\Sigma}_{t+\delta t|t}) \\ P_{t+\delta t|t+\delta t} &= \mathcal{N}(\mathbf{x}_{t+\delta t|t+\delta t}; \boldsymbol{\mu}_{t+\delta t|t+\delta t}, \boldsymbol{\Sigma}_{t+\delta t|t+\delta t}) \end{aligned}$$

Motion and observation models are also multivariate normal distributions with the means being linear function of the conditioning state:

$$\begin{aligned} P(\mathbf{x}_{t+\delta t}|\mathbf{x}_t) &= \mathcal{N}(\mathbf{x}_{t+\delta t}; \mathbf{F}_{\delta t}\mathbf{x}_t, \mathbf{Q}_{\delta t}) \\ P(\mathbf{o}_t|\mathbf{x}_t) &= \mathcal{N}(\mathbf{o}_t; \mathbf{H}\mathbf{x}_t, \mathbf{R}) \end{aligned}$$

Under these assumptions, the observation model is conjugate to the predicted state density and the marginalization and update step can be expressed in closed form update equations. The parameters of $P_{t+\delta t|t}$ are calculated as

$$\begin{aligned} \boldsymbol{\mu}_{t+\delta t|t} &= \mathbf{F}_{\delta t}\boldsymbol{\mu}_{t|t} \\ \boldsymbol{\Sigma}_{t+\delta t|t} &= \mathbf{F}_{\delta t}\boldsymbol{\Sigma}_{t|t}\mathbf{F}_{\delta t}^T + \mathbf{Q}_{\delta t} \end{aligned}$$

The parameters of $P_{t+\delta t|t+\delta t}$ are then calculated as

$$\begin{aligned} \mathbf{S} &= \mathbf{H}\boldsymbol{\Sigma}_{t+\delta t|t}\mathbf{H}^T + \mathbf{R} \\ \mathbf{K} &= \boldsymbol{\Sigma}_{t+\delta t|t}\mathbf{H}^T\mathbf{S}^{-1} \\ \boldsymbol{\mu}_{t+\delta t|t+\delta t} &= \boldsymbol{\mu}_{t+\delta t|t} + \mathbf{K}(\mathbf{o}_{t+\delta t} - \mathbf{H}\boldsymbol{\mu}_{t+\delta t|t}) \\ \boldsymbol{\Sigma}_{t+\delta t|t+\delta t} &= (\mathbf{I} - \mathbf{K}\mathbf{H})\boldsymbol{\Sigma}_{t+\delta t|t} \end{aligned}$$

This approach is known as the Kalman Filter [39].

The other extreme is to resort to a pure ensemble based representation for $P_{t|t}$ and $P_{t+\delta t|t}$. Then, we are free to use an arbitrary, often stochastic, function $\mathcal{F}_{\delta t} : \mathbf{x}_t \to \mathbf{x}_{t+\delta t}$ to transform the ensemble representing $P_{t|t}$ into an ensemble representing $P_{t+\delta t|t}$, the so-called proposal distribution. We can then use an arbitrary observation likelihood function $P(\mathbf{o}_{t+\delta t}|\mathbf{x}_{t+\delta t})$ to generate importance weights for the samples in the proposal distribution which is in turn re-sampled according to these importance weights to obtain an ensemble that represents $P_{t+\delta t|t+\delta t}$ and the process repeats. This so-called particle filter [40] does not make any assumption on the state space, i.e. one could use sets instead of vectors as representations of the state.

4.6.2.7 **State**:
State Density and State Density Updates through Bayesian Filtering For a traffic scenario tracking and prediction system, we focus on the traffic participants only. Though Bayes filtering can be and is being applied in the estimation of the current state of the static environment as well, the complexity of predicting its future over the time spans we are interested in is limited.

Prior knowledge commonly enters the modeling through the definition of the state space. For example the non-holonomic constraints in the movement of a vehicle with Ackerman steering are introduced by resorting to the bicycle model and its kinematic state space of $\mathbf{x}_t = [x, y, v, a, \psi, \delta]$ with position x, y, forward velocity and acceleration v and a, heading angle ψ and steering angle δ [41]. This is a perfect state space if one wants to *drive* the vehicle from one state to another, i.e. solve a control problem. The situation is markedly different though when one aims to *predict* where a vehicle is going. Consider the following situation in a highway on-off ramp in Figure 22. Two vehicles are seen in different lanes with otherwise identical kinematic states. The road structure makes it clear that each of them has two possible options - enter or leave the highway which results in a total of four different ways of how the entire scene could evolve. Assuming that non of the vehicles actively signals its intention, an autonomous vehicle behind these two traffic participants would thus have to deal with 4 potential future developments of the scenario in front.

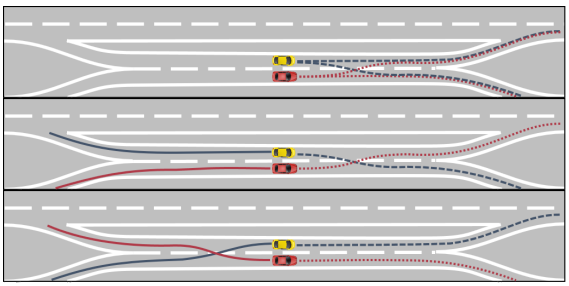


Fig. 22: Top: Given only kinematic states, each vehicle has 2 plausible path options, resulting in a total of 4 equally likely future scenarios for the traffic scene. Middle and Bottom: With past trajectories given, the uncertainty of future scenarios may be largely resolved.

Clearly, kinematic states capture the physical aspects of vehicle motion, but they do not capture the fact that autonomous agents move with goals and intent. Hence, in order to be the sole generator of dynamics, we would have to augment the kinematic state with some descriptors of driver intent, such as maneuver intentions [42, 43]. Such intent is not directly measurable and has to be inferred from past observations. However, that is a challenging problem as it is not clear how to describe the space of maneuver intentions. Should maneuver intentions be mutually exclusive and collectively exhaustive? Does that mean I cannot make a lane change while taking a turn? Also, does this apply to all traffic participants? What are the maneuvers of a cyclist or a pedestrian?

Since all the obtainable information about a driver's intent must be in its past behavior, i.e. its trajectory, it is tempting to simply maintain a list of past kinematic states and infer driver intent on the fly based on this "historic" data. Note that that a list of past states is not the same as list of past observations - state tracking algorithms disentangle the uncertain data associations that arise in multi-object tracking giving unique physical objects unique tracking ids without which single object trajectories cannot be formed.

While promising, this idea has the obvious drawback of being both memory and compute intensive. This is particularly the case if tracking multiple objects with multi-hypothesis tracking algorithms which potentially maintain hundreds of data association hypotheses in order to be able to track effectively through clutter and occlusion [44, 45, 46]. Since each data association hypothesis effectively represents a different history, one would have to maintain hundreds of historic trajectories.

We take up the idea of taking the trajectory of an object as the basis for predicting its future motion. However, instead of maintaining a list of kinematic states, we propose to use an object's past trajectory over a constant time horizon Δt as its *state*. We introduce a parametric representation for this trajectory with small memory footprint independent of Δt and corresponding linear motion and observation models that allow to estimate the parameters of this representation using closed form Kalman update equations without the need for approximations. The small memory footprint makes trajectory tracking possible to be used in conjunction with multi-object trackers in real-time capable systems run on embedded hardware.

To be precise, our state $\mathbf{x}_t$, will at any moment t describe the past trajectory of the object over a *fixed*

time horizon Δt from $t - \Delta t$ to t.

As an object moves through a d-dimensional space, we model the pose of the object as a function of time $f_j(t)$ in each dimension $j \in \{1..d\}$. In what follows we restrict ourselves to understand pose as position, but this could be extended.

We recall the Taylor series expansion with its Lagrange remainder:

$$f(t_0 + \delta t) = \sum_{k=0}^{n} \frac{1}{k!} f^{(k)}(t)\Big|_{t=t_0} \delta t^k + \frac{1}{(n+1)!} f^{(n+1)}(t)\Big|_{t=t^*,\ t^* \in [t_0, t_0+\delta t]} \delta t^{n+1}$$

The remainder tells us that the error we maximally make when approximating a trajectory with polynomials of order n is bounded by the maximum of the $n+1$ derivative with respect to time of the function. Fortunately for us, we can expect the higher derivatives to be small because all traffic participants avoid jerky movements and physics sets limits on the rate of change in their coordinates time that traffic participants can muster [47, 48, 49]. In fact, it is the hallmark of experienced drivers that their maneuvers are smooth and *thus* predictable. In normal traffic, almost all trajectories are smooth and the ones that are not will be detected as anomalous and dealt with in a separate manner. Human drivers tend to keep greater safety distances to erratically moving traffic participants precisely because they know they cannot predict well where such drivers are going.

We now proceed as follows. We choose a set of $n+1$ *fixed* basis functions $\phi_n(\tau) : \mathbb{R} \to \mathbb{R}$ to describe the trajectory in each of the d dimensions of space. We differentiate in $\tau \in \mathbb{R}$ a re-scaled time variable that denote time along the tracked trajectory with $\tau = 0$ corresponding to $t - \Delta t$ and $\tau = 1$ corresponding to t. It is convenient to introduce $\mathbf{\Phi}(\tau)$ as the vector of basis functions evaluated at τ:

$$\mathbf{\Phi}(\tau) = \begin{bmatrix} \phi_0(\tau) & \phi_1(\tau) & \phi_2(\tau) & \cdots & \phi_n(\tau) \end{bmatrix}$$

Next, we introduce our state vector $\mathbf{x}_t \in \mathbb{R}^{(n+1)d}$. Together with the fixed basis functions, we can generate the coordinates of any point $\mathbf{c}_t(\tau) \in \mathbb{R}^d$ on the trajectory from

$$\mathbf{c}_t(\tau) = (\mathbf{\Phi}^T(\tau) \otimes \mathbf{I}_d)\mathbf{x}_t$$

According to our definitions, the objects current position will be $\mathbf{c}_t(1)$ and the object's position at $t - \Delta t$ will be $\mathbf{c}_t(0)$.

It is instructive to interpret the entries in our state vector $\mathbf{x}_t$ as *control points*. Separated into individual coordinates and basis vectors $\mathbf{e}_i$ of the d dimensional space we have (dropping the index t for brevity):

$$\mathbf{c}(\tau) = \sum_{i=1}^{d} \sum_{k=0}^{n} \phi_k(\tau) x_{kd+i} \mathbf{e}_i = \sum_{k=0}^{n} \phi_k(\tau) \mathbf{p}_k$$

Thus, our trajectory is a weighted combination of $(n+1)$ control points with the basis functions as weights that vary with τ. So the first d entries in $\mathbf{x}_t$ correspond to $\mathbf{p}_0$, the second d entries correspond to $\mathbf{p}_1$, etc. If we arrange the control points as the rows of a matrix $\mathbf{P} \in \mathbb{R}^{(n+1)\times d}$ we can write most compactly:

$$\mathbf{c}(\tau) = \mathbf{\Phi}^T(\tau)\mathbf{P}$$

We thus have a compact representation of a trajectory in terms of its control points. We stress the importance of having a parameterization that has spatial semantics: Since trajectory prediction and scenario prediction in particular is about the interaction of curves extending through time and space with purely spatial features. It appears that a common footing will provide the basis of effectively modeling the interactions of trajectories and the static environment.

Our representation affords a few more insights once we take a closer look at specific basis functions which we do next.

4.6.2.8 **Basis Functions**:

A natural choice of basis functions are of course monomials $\phi_k(\tau) = \tau^k$ as motivated by the Taylor expansion. This choice is by no means unique, but provides a good starting point. The interpretability of the control points is improved when using Bernstein Polynomials

$$\phi_k(\tau) = \begin{pmatrix} n \\ k \end{pmatrix} \tau^k (1-\tau)^{(n-k)}$$

which are simply linear combinations of ordinary monomials and thus are an equivalent representations. They have the advantage that $\mathbf{p}_0$ and $\mathbf{p}_n$ form the endpoints of the trajectory, $\mathbf{p}_1 - \mathbf{p}_0$ is tangent

to the trajectory at $\tau = 0$, etc. With Bernstein Polynomials as basis functions, we are effectively using Bézier curves to represent our trajectories [50, 51, 52, 53].

If $\mathbf{\Phi}_M(\tau)$ is the vector of the standard monomial basis functions, we can obtain $\mathbf{\Phi}_B(\tau)$, the vector of Bernstein Polynomial basis functions, from:

$$\mathbf{\Phi}_B^T(\tau) = \mathbf{\Phi}_M^T(\tau)\mathbf{M}$$

where the coefficient matrix $\mathbf{M}$ is triangular with entries [54]:

$$\boldsymbol{M}_{ij} = \begin{cases} \binom{n}{j-1}\binom{n-j+1}{i-j}(-1)^{(i+j) \mod 2} & \text{if } j \leq i \\ 0 \text{ otherwise} \end{cases}$$

Since our basis functions are fixed, we also have a chance to learn them from data. If the intervals at which we need to evaluate our trajectories are fixed because we have a constant cycle frequency in the tracking updates, then we may even choose to learn the basis functions at a finite set of discrete sampling points τ_i.

These would then constitute a fully interpretable and inspectable set of parameters.

If the basis functions are all non-negative and scaled to lie between zero and one, then we also constrain all representable trajectories to the convex hull of the control points. Non-negativity and scaling may be enforced during learning if one is to choose so.

4.6.2.9 **State Density**:

With our state vector $\mathbf{x}_t$ from $\mathbb{R}^{(n+1)d}$ it is natural to assume a multivariate Gaussian state density [52]. Figure 23 shows an example for $n = 3$, $d = 2$ with Bernstein Polynomials as basis functions. The control points $\mathbf{p}_k$ are drawn at their mean values with corresponding 95% confidence ellipses around them. We further denote the mean trajectory as solid line and show a few samples from the distribution as dashed lines.

It should be noted that the covariance matrix of our state density is full, i.e. we allow explicit correlations between dimensions on top of the correlations between the coefficients of our basis functions along a single dimension [55].

4.6.2.10 **Prior**:

We set out to model smooth and comfortable trajectories [56, 57, 53, 48]. This can be enforced by putting a prior on the entries of the state vector. Such a prior can either be learned from data or set by hand. As a general rule of thumb, restricting the variance of fast changing basis functions, e.g. higher order monomials, results in smoother trajectories. Figure 24 shows an example of this, comparing random draws from a zero mean multivariate Gaussian with different covariance matrices. We should note that covariance structure is a very powerful tool here, as it allows to model typical driving behaviors such as the fact that directional changes are small at large velocities through negative correlations between velocities in accelerations in different spatial dimensions.

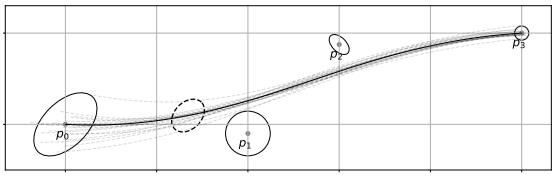


Fig. 23: Example of a trajectory representation with $n = 3, d = 2$. The solid line corresponds to the mean trajectory and the dashed lines are samples from the density. Parameters, i.e. control points, are indicated at the mean values and with 95% confidence intervals. The dashed ellipse encloses a 95% confidence region for the position of the object at $\tau = 0.2$.

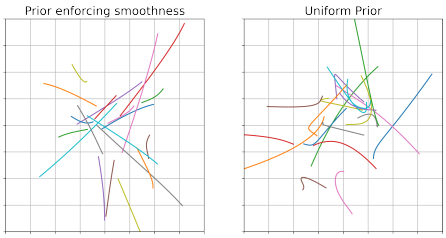


Fig. 24: Samples from different prior distributions that favor smoother and jerkier trajectories

4.6.2.11 **Observation Model**:

Our formalism allows to make observations of the trajectory at any point. However, we are naturally

most interested in the observation model at $\tau = 1$, the end of trajectory at t. The natural observables are of course $\mathbf{c}_t(\tau)$ and its derivatives with respect to time, i.e. velocities and accelerations. Due to our re-scaling, we have $\mathrm{d}t = \Delta t \mathrm{d}\tau$:

$$\left.\frac{\mathrm{d}^n}{\mathrm{d}t^i}\mathbf{c}_t(\tau)\right|_{\tau=1} = \underbrace{\frac{1}{(\Delta t)^n}\left(\left.\frac{\mathrm{d}^n}{\mathrm{d}\tau^n}\mathbf{\Phi}^T(\tau)\right|_{\tau=1} \otimes \mathbf{I}_d\right)}_{\mathbf{H}_n}\mathbf{x}_t$$

This amounts to a constant observation matrix for every value of τ that can be easily combined. For example, let us assume we are observing an object's position x, y and velocity v_x, v_y organized in an observation vector $\mathbf{o}_t = [x, y, v_x, v_y]$, the corresponding linear observation model $\mathbf{H}$ is then given by stacking the rows of $\mathbf{H}_0$ and $\mathbf{H}_1$ from above. The corresponding observation noise $\mathbf{R}$ can then be used to reflect corresponding measurement noise.

4.6.2.12 **Motion Model**:

The next component we need in our filtering architecture is the motion model. We will consider two ways to derive a linear model. First, let us consider the case in which we can find a linear derivative operator $\mathbf{D}$ for our vector of basis functions:

$$\dot{\mathbf{\Phi}}^T(\tau) = \mathbf{\Phi}^T(\tau)\mathbf{D}$$

For the monomial basis functions, this derivative operator is obviously

$$\mathbf{D}_M = \begin{bmatrix} 0 & 1 & 0 & 0 & \cdots \\ 0 & 0 & 2 & 0 & \cdots \\ 0 & 0 & 0 & 3 & \cdots \\ & & \cdots & & \ddots \end{bmatrix}$$

We now observe the associativity in:

$$\begin{aligned} \dot{\mathbf{c}}(\tau) &= \dot{\mathbf{\Phi}}^T(\tau)\mathbf{P} = (\mathbf{\Phi}^T(\tau)\mathbf{D})\mathbf{P} \\ &= \mathbf{\Phi}^T(\tau)(\mathbf{D}\mathbf{P}) = \mathbf{\Phi}^T(\tau)\dot{\mathbf{P}} \end{aligned}$$

which gives us a linear motion model for the control points $\dot{\mathbf{P}} = \mathbf{D}\mathbf{P}$ and consequently for our state vector:

$$\frac{\mathrm{d}}{\mathrm{d}t}\mathbf{x} = \frac{1}{\Delta t}(\mathbf{D} \otimes \mathbf{I}_d)\mathbf{x} \tag{3}$$

for a given $\delta t = \Delta t \delta\tau$ this can be integrated analytically as a matrix exponential to yield:

$$\begin{aligned} \mathbf{F}_{\delta t} &= \exp\left(\frac{\delta t}{\Delta t}(\mathbf{D} \otimes \mathbf{I}_d)\right) \\ \mathbf{x}_{t+\delta t} &= \mathbf{F}_{\delta t}\mathbf{x}_t \end{aligned}$$

For Bernstein Polynomials, we can form $\mathbf{D}_B = \mathbf{M}^{-1}\mathbf{D}_M\mathbf{M}$ via a similarity transform.

The second way of generating a motion model is derived from a fit procedure. Assume at time t, we have $m \geq n+1$ samples of a trajectory $\mathbf{c}_t(\tau_i)$ along a trajectory at different times $\tau_k,\ k \in \{1, .., m\}$ between $\tau_0 \geq 0$ and $\tau_m \leq 1$. We arrange these samples as the rows of an $m \times d$ matrix $\mathbf{C}_t$. We now form an $m \times (n+1)$ matrix $\mathbf{B}$ so that row k of $\mathbf{B}$ corresponds to $\mathbf{\Phi}(\tau_k)$. Then, we can estimate the control points as a Bayesian least square fit to the samples of the trajectory:

$$\mathbf{P}_t = (\mathbf{B}^T\mathbf{R}^{-1}\mathbf{B} + \mathbf{\Sigma}_P^{-1})^{-1}\mathbf{B}^T\mathbf{R}^{-1}\mathbf{C}_t$$

Here $\mathbf{\Sigma}_P$ is the covariance matrix of a possible zero mean Gaussian prior and $\mathbf{R}$ the Gaussian observation noise covariance. Note that a zero mean prior is a natural assumption if one of our basis functions is constant as is the case for both $\mathbf{\Phi}_M$ and correspondingly $\mathbf{\Phi}_B$.

The way to propagate a trajectory forward in time now is the following: From the control points $\mathbf{P}_t$, we generate samples along the mean of the trajectory at $m = n+1$ equal spaced $\tau'_k = \delta t/\Delta t + k/n(1 - \delta t/\Delta t),\ 0 \leq k \leq n$ and then re-estimate the control points from these samples pretending they were obtained at $\tau_k = \tau'_k - \delta t/\Delta t$. To do this, as before, we construct the $(n+1)\times(n+1)$ matrices $\mathbf{B}$ and $\mathbf{B}'$ from $\mathbf{\Phi}(\tau_k)$ and $\mathbf{\Phi}(\tau'_k)$, respectively. We then get transformed control points from

$$\mathbf{P}_{t+\delta t} = (\mathbf{B}^T\mathbf{B} + \mathbf{\Sigma}_P^{-1})^{-1}\mathbf{B}^T\mathbf{B}'\mathbf{P}_t$$

Since we are estimating the new control points from samples of the mean, we don't need to consider the observation covariance matrix $\mathbf{R}$ in this expression. Without a prior, this formula reduces to

$$\mathbf{P}_{t+\delta t} = \mathbf{B}^{-1}\mathbf{B}'\mathbf{P}$$

and it should be noted this formula is consistent with our earlier first approach if a linear derivative operator exists:

$$\mathbf{B}^{-1}\mathbf{B}' = \exp\left(\frac{\delta t}{\Delta t}\mathbf{D}\right)$$

and we thus have a linear motion model $\mathbf{F}_{\delta t}$ for the control points that only depends on our choice of basis functions and correspondingly for the state vector we find

$$\mathbf{F}_{\delta t} = ((\mathbf{B}^T\mathbf{B} + \mathbf{\Sigma}_P^{-1})^{-1}\mathbf{B}^T\mathbf{B}') \otimes \mathbf{I}_d \quad (4)$$
$$\mathbf{x}_{t+\delta t} = \mathbf{F}_{\delta t}\mathbf{x}_t \quad (5)$$

Figure 25 illustrates this on a cubic Bézier curve and exemplifies the differences of the motion model with and without the use of a prior.

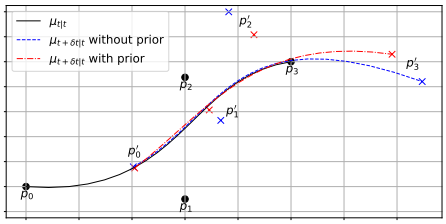


Fig. 25: Example of the action of the motion model with $\delta t = 0.3\Delta t$. Note how the motion model without prior follows the trajectory exactly for all points in the past, while the model with prior deviates slightly from the past estimation.

It may be of desirable to have certain flexibility in the dependence of $\mathbf{F}_{\delta t}$ on δt. For example, it appears natural that the duplicate application of the motion model $\mathbf{F}_{\delta t}$ results in the same prediction as the single application of $\mathbf{F}_{2\delta t}$. Generically, let $\mathcal{F}_{\delta t} : \mathbf{x}_t \to \mathbf{x}_{t+\delta t}$ be a motion model, we then have the following *equivariance constraint*:

$$\mathcal{F}_{\delta t}(\mathcal{F}_{\delta t'}(\mathbf{x}_t)) = \mathcal{F}_{\delta t+\delta t'}(\mathbf{x}_t)$$

This is automatically satisfied by $\mathcal{F}_{\delta t}(\mathbf{x}_t) = e^{\delta t\mathbf{D}}\mathbf{x}_t$ and for motion models resulting from linear differential equations $\dot{\mathbf{x}} = \mathbf{D}\mathbf{x}$.

It is clear that not all linear motion models result from integrating linear differential equations as we cannot always find a linear differential equation of the form (3) corresponding to (4). Hence, using a fixed prior already violates this equivariance constraint. Still, it appears to be a very natural constraint on motion models.

4.6.2.13 **Ego Motion Compensation**: Motion Model The last aspect of our development of a state representation in the form of trajectories is ego motion compensation. Due to sensor movement, the frame of reference $\mathcal{O}_{t+\delta t}$ in which observations are obtained is generally different from $\mathcal{O}_t$ which is used as the frame of reference for the current state estimate $P_{t|t}$. For situation interpretation, it is convenient to maintain an alignment of the current state estimate and the current observer position. Hence, we either transform $P_{t|t}$ to $\mathcal{O}_{t+\delta t}$ before forming $P_{t+\delta t|t}$ in the prediction step, or we perform the coordinate transformation afterwards.

We consider the situation in 2D. Between time t and time $t + \delta t$, the observer has been translated $\Delta\mathbf{o} = [\Delta x, \Delta y]$ and rotated by $\delta\psi$ such that the observer's frame of reference is now $\mathcal{O}'$ in which measurements are taken and all trajectories must be expressed.

The rotation is characterized by a rotation matrix

$$\mathbf{R} = \begin{bmatrix} \cos(\delta\psi) & \sin(\delta\psi) \\ -\sin(\delta\psi) & \cos(\delta\psi) \end{bmatrix}$$

and any control point would be transformed as $\mathbf{p}_i \leftarrow (\mathbf{p}_i - \Delta\mathbf{o})\mathbf{R}^T$.

Since the covariance of our trajectory estimate is only affected by the rotation of the coordinate system, we have

$$\mathbf{\Sigma}_t \leftarrow (\mathbf{I}_n \otimes \mathbf{R})\mathbf{\Sigma}_t(\mathbf{I}_n \otimes \mathbf{R})^T$$

In order to compensate the mean of our estimate in a linear transform, we have to introduce homogeneous coordinates, i.e. we will introduce an additional dimension to our control points that is identically 1. The homogenized $\boldsymbol{\mu}_t$ then reads:

$$\boldsymbol{\mu}_t^h = [\mu_{0,x}, \mu_{0,y}, 1, \cdots, \mu_{n,x}, \mu_{n,y}, 1]$$

We then introduce a transformation matrix $\mathbf{T} \in \mathbb{R}^{d\times(d+1)}$ that a mediates both the translation and rotation necessary in the change of coordinates:

$$\mathbf{T} = \begin{bmatrix} \mathbf{R} & -\mathbf{R}\Delta\mathbf{o} \end{bmatrix}$$

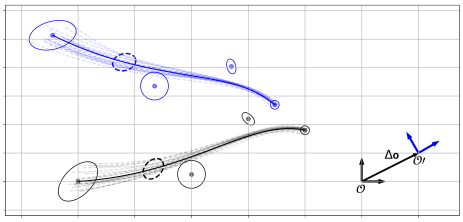


Fig. 26: Illustration of ego compensation for the trajectory representation. If the observer moves from $\mathcal{O}$ to $\mathcal{O}'$, the black trajectory will appear as the blue trajectory seen from $\mathcal{O}'$.

We obtain the transformed mean vector from the homogenized mean vector via

$$\boldsymbol{\mu}_t \leftarrow (\mathbf{I}_n \otimes \mathbf{T})\boldsymbol{\mu}_t^h$$

It is worth pointing out that the control points transform also via homogeneous coordinates - as

$$\mathbf{P} \leftarrow \mathbf{P}^h\mathbf{T}^T$$

where $\mathbf{P}^h$ is the homogenized matrix of control points, i.e. the matrix $\mathbf{P}$ with an additional column. Figure 26 illustrates the situation on our example from Figure 23 and shows how all covariances and uncertainties transform easily under sensor movement. This step is crucial as situation interpretation is much easier in the coordinate frame of the vehicle and the uncertainties play a key part in situation interpretation.

We already pointed out that ego motion compensation can be performed before or after the prediction step. If we assume the motion of traffic participants independent from that of the observer, we can derive an equivariance constraint: Let $\mathcal{T}$ be the function that changes the frame of reference from $\mathcal{O}_t$ to $\mathcal{O}_{t+\delta t}$ and let $\mathcal{F} : P_{t|t} \rightarrow P_{t+\delta t|t}$ be the function that performs prediction step. Consequently, we must have the following *equivariance constraint*:

$$\mathcal{T}(\mathcal{F}(P_{t|t})) = \mathcal{F}(\mathcal{T}(P_{t|t}))$$

in other words, the operations of ego-compensation and the prediction step commute. For the linear motion models and linear operations to change coordinates, this follows from associativity as seen e.g. in this formulation:

$$\mathbf{P}_{t+\delta t} = \mathbf{F}_{\delta t}\mathbf{P}_t^h\mathbf{T}^T$$

If the predicted motion of other traffic participants is not independent of the motion of the observer, then this equivariance is no longer valid. It is then also preferred to first perform ego motion compensation and then the prediction step as this allows to inform the motion model with the true position of the observer at $t + \delta t$.

4.6.2.14 **Summary**:

We have introduced a parametric representation for object trajectories as a Markov state for object tracking together with corresponding linear motion and observation models that allow closed form Kalman filtering under a Gaussian density approximation. Albeit we have illustrated and motivated these concepts primarily with vehicles, they are by no means limited to represent vehicle trajectories. Rather, they apply to all physical traffic participants that cannot change their state of motion arbitrarily fast [47, 57, 48, 49]. The representation is not limited to past trajectories but naturally extends to future trajectories as well. Within AP1.2 we further show how this representation can be used in trajectory prediction tasks and how it can be integrated into multi-object multi-hypothesis trackers to represent consistent probabilistic multi-modal distributions over the future of entire traffic scenarios.

4.7 Deutsches Forschungszentrum für Künstliche Intelligenz GmbH (DFKI)

Topic overview:

Use Cases (Scenario)
UC 2

Knowledge Source
Images

Knowledge Formalization & Representation
Ontology, Knowledge Graph

Conformity Check
No involvement

4.7.1 Introduction

A traffic scene is a complex environment where multiple objects are simultaneously present on the road like pedestrians, vehicles cyclists, etc. These objects are continuously interacting with each other, which makes a traffic scene even more complicated. Subsequently, it is very crucial to make sure that the ego vehicle is safe in a traffic environment. The term "safe" refers to the scenario where the ego vehicle avoids any potential collision with a pedestrian or any other vehicle on the road while successfully maneuvering through the traffic.

4.7.2 Use Cases

Our work in these work packages is related to UC2 which is about lane change. We contribute here by evaluating the existing traffic situation from the point of view of safety. So that a safe and collision-free lane change maneuver could be performed.

4.7.3 Knowledge sources

The video frames from a camera mounted on the ego vehicle will serve as the primary source of knowledge. These images contain information about the traffic scene around the ego vehicle. Detected pedestrians in each traffic scene will constitute the direct knowledge. In addition, we will estimate implicit information like speed, direction, and relative location of the pedestrian. The location of a pedestrian can be found in a single frame using the center of a detected pedestrian. However, for estimating speed and direction, we plan to employ multiple consecutive video frames. Additionally, the data from different sensors installed on the ego vehicle will serve as another knowledge source.

4.7.4 Representation/Formalization concept

Safety plays a crucial role in the smooth flow of traffic on the road. Vehicles constantly perform different maneuvers to navigate through the traffic. With the increasing number of vehicles and other participants on the road, it is becoming increasingly important to continuously estimate ego vehicle safety. For this purpose, the information collected from all traffic participants in a traffic scene is decisive.

In this context, we plan to use two different kinds of knowledge based on the accessibility of knowledge. One is unequivocal knowledge and the other is implicit knowledge. As the names suggest, the first type of knowledge is observable and is readily available. The more concrete example of such knowledge is video frames, where all objects present in a traffic scene are visible. In real-life scenarios, there could be dozens of objects present in an image from a traffic scene on a road. Where all objects are constantly in motion and our task is to safely maneuver through this complex environment without any collision. The simultaneous interaction of the existing objects in a traffic scene represents the complexity of the challenge and the task at hand. We can certainly extract such knowledge and use it as our baseline.

The second type of knowledge we are interested in is implicit knowledge, which is not noticeable in a traffic scene image. However, it can be implied given the set of existing objects is provided along with the traffic imagery. A more specific example of such knowledge is attributes related to the objects in a traffic environment. Such attributes include the position, speed, and direction of an existing object like a pedestrian, etc. We can extract such knowledge by processing a set of consecutive video frames that can distinctly estimate these attributes for each of the traffic participants.

Once we have data related to all the objects and their complex interactions in a traffic scene, only then we can ensure the safe mobility of the ego vehicle through such a complex environment. It also serves as a crucial milestone for achieving autonomous driving. Subsequently, we plan to formalize all the knowledge from a traffic scene in the form of a knowledge graph.

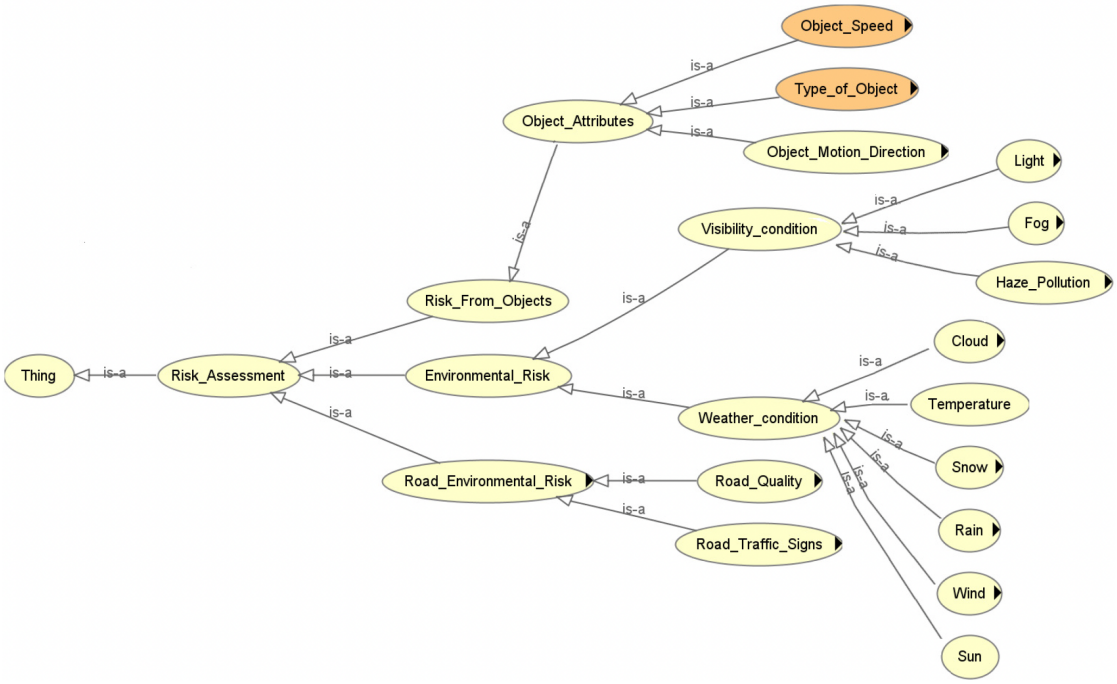


Fig. 27: Selected Baseline Ontology.

Our pipeline begins with continuous video imagery from the camera installed on an ego vehicle. With this imagery, we can observe the traffic situation around ego vehicles. Firstly, we plan to detect all relevant objects in the traffic imagery. These objects include pedestrians, cyclists, traffic signals, and other motorists on the road. For this purpose, we can employ any existing object detection model like Mask RCNN, Faster-RCNN, etc. An object detection model can not only detect the presence of the relevant objects in a traffic scene but also detect their position in the image. The detected objects are enclosed in bounding boxes that provide us an adequate idea of the relative position of other traffic participants. However, this knowledge is insufficient to ensure the safety of a traffic scene since we have relative information about other traffic participants in 2D space. Therefore, we lack 3D information like relative distance from the ego vehicle as a single frame is insufficient to estimate 3D attributes of an object in a traffic scene image.

To overcome this challenge, we make use of multiple consecutive video frames to estimate the relative speed and direction of a detected object in a traffic scene. The distance traveled by an object in consecutive frames is measured in pixels. We perform this estimation individually for each of the detected objects in the traffic imagery. Once we have all the knowledge from direct or indirect sources, we need to formalize this knowledge such that it can be directly plugged into any other pipeline of autonomous driving.

The first step towards knowledge formalization is to formulate the complex interactions between traffic participants on the road in a defined structure. Ontologies serve as a proficient way to define these complex interactions effectively. There are several existing ontologies related to autonomous driving. However, most of those ontologies lack the variety of objects, attributes, and relations relevant to our use case. It is a challenging task to find an ontology that perfectly fits any specific use case. Therefore, we selected an ontology that will serve as a baseline ontology for our work. We plan to customize the

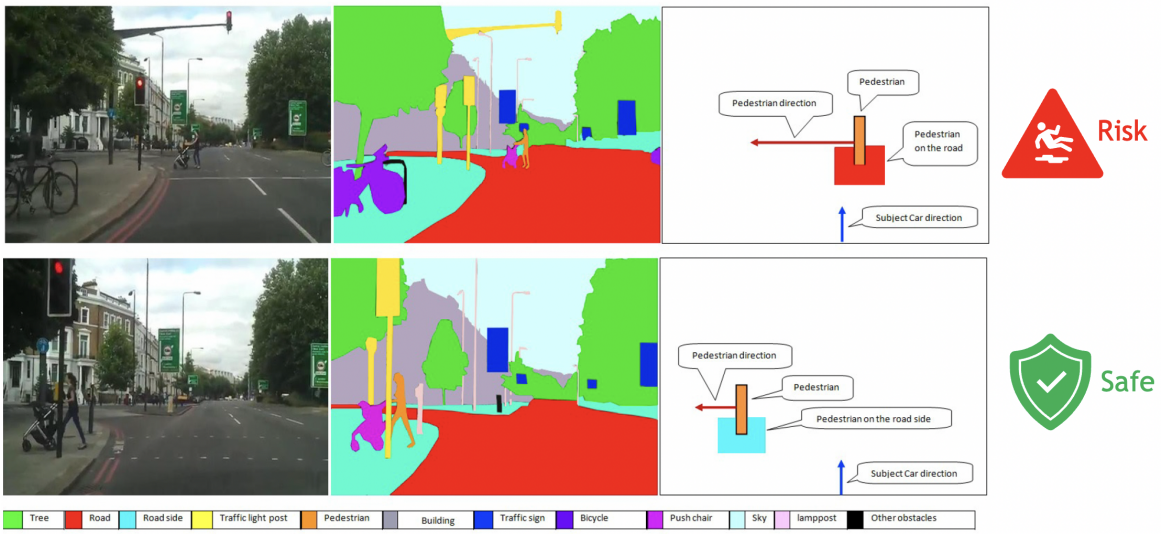


Fig. 28: Example of Traffic Risk Assessment Scenarios.

selected ontology by adding missing attributes or objects relevant to traffic scene safety.

Fig 27 shows the selected baseline ontology. It can be observed that the risk in a traffic scene is estimated using a variety of different aspects. Some of which are related to detected objects while others occur independently for example weather and road conditions etc. The nodes named Object_Speed, Type_of_Object, and Object_Motion_Direction are directly related to the detected objects. The names of the nodes are self-descriptive as they represent the speed, type, and direction of a detected object respectively. It can also be observed that the selected ontology is missing crucial entities like a traffic signal, road signs, etc. which also play an important role in the safety of a traffic scenario. We plan to include all missing entities and attributes to make it a perfect fit for our use case.

Lastly, after collecting all the data from different sources and estimating implicit knowledge, we plan to plug all the information into the selected ontology to generate a knowledge graph. This knowledge graph is then used to estimate the safety of a given traffic scenario. Fig 28 shows an example of a safe and unsafe scenario. The example on the top represents an unsafe scenario, since a pedestrian is on the road while the ego vehicle is approaching. However, in the bottom scenario, we can see that the pedestrian is on the sidewalk therefore making the given scenario safe for the approaching ego vehicle.

4.8 Deutsches Zentrum für Luft- und Raumfahrt e.V. (DLR)

Use Cases (Scenario)
UC 2
Knowledge Source
Object co-occurrence relationship
Knowledge Formalization & Representation
Knowledge Graph
Conformity Check
No involvement

4.8.1 Introduction

With the aim of developing Hybrid - Deep Learning (H-DL) system for holistic situation understanding for traffic evolution prediction, DLR in AP 1.2 focused on primitive task of traffic elements detection on images, specifically traffic sign detection. World knowledge (e.g. co-occurrence relationship and spatial relationship) is integrated with existing object detection framework for detecting traffic signs on the camera images.

4.8.2 Use Cases

DLR intends to support in addressing complex lane change prediction through holistic scene understanding. Scene-entity relationship place vital role in predicting the evolution of traffic, therefore DLR is researching towards possibility of incorporating world-knowledge (traffic elements e.g. traffic sings, road attributes such as lane marking types) with traffic norms (e.g. StVo rules) for predicting the evolution of traffic for motorized objects and predicting lane change behavior for AD vehicle in UC2.

4.8.3 Knowledge formulation and incorporation

In respect to knowledge representation DLR make use of object detector paradigms that treat object bounding box regression and classification of each region proposal separately. Within [58], the author exploited three different object commonsense knowledge a) shared attribute knowledge; b) pairwise relationship knowledge; c) spatial layout. Inspired from this work, we consider "pairwise relationship knowledge" for integrating with Faster-RCNN [59] object detector pipeline. Wherein, specific knowledge module in the object detector pipeline learns adaptive context connections for each pairwise regions by using "pairwise relationship knowledge" graph as external supervision. The pairwise relationship knowledge is statically observed from the training data samples which is inherent and implicitly hidden.

As shown in 29: "Explicit pairwise relationship" module incorporates adaptive region-to-region pairwise relationship knowledge without being summarized by the human. visual region features extracted on input images through region proposal network. With the region feature. Based on the region features, "Explicit knowledge" module in 29 builds an adaptive region-to-region undirected graph $\hat{G}$: $\hat{G} =< N, \hat{\varepsilon} >$, where N defines the region nodes and $e_{i,j} \in \varepsilon$ defines the pairwise relationship (i.e co-occurence relationship) between two nodes i, j. Enhanced features from the knowledge module are concatenated with based region features and fed into the bounding-box regression layer and classification layer to obtain detection results.

For more explicit information on this topic we refer to the DLR contribution to the KI Wissen first Deliverable AP 1.2.

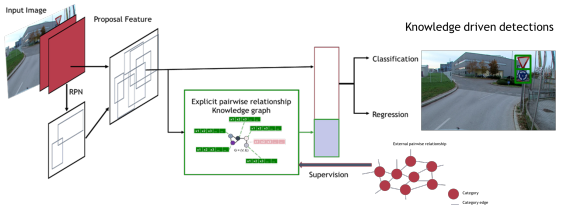


Fig. 29: Overview of Knowledge directed traffic sign detection.

4.9 Elektronische Fahrwerksysteme GmbH (EFS)

Topic overview:

Use Cases (Scenario)
UC 2

Knowledge Source
Vehicle Dynamics (Laws of Physics)

Knowledge Formalization & Representation
Structural Causal Model (SCM)

Conformity Check
Answers to Causal Queries / Sensitivity Analysis

4.9.1 Introduction

Besides our goal of integrating physical-mathematical knowledge of the trajectory evolution process into autonomous vehicles, we are interested in exploiting this knowledge for checking whether an AI-component's decision conforms with this knowledge and whether this decision might entail negative consequences, e.g., in terms of potential accidents. Therefore, it is unsurprising that our proposed knowledge integration & conformity concepts are closely intertwined with each other in the sense that the knowledge is both integrated during an AI's training process and moreover used to check the validity of an AI's consumed input data as well as its predictions with regard to that knowledge. A high-level overview of the interplay between our concepts is depicted in Fig. 30. As we will discuss

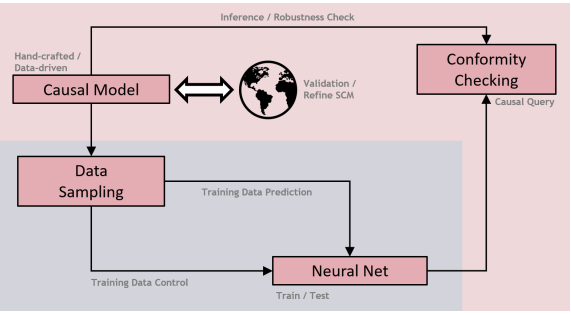


Fig. 30: High-Level Overview of our Knowledge Integration (gray shaded area) & Conformity (red shaded area) Concepts.

in the following, a large part of our knowledge conformity concept relies on the *Structural Causal Model (SCM)* framework [60]. Based on this, we represent the physical-mathematical knowledge, that we focus on, in terms of an SCM. Knowledge formalized in this way is then used to generate cause-effect-aware monitors of the AI. Eventually, these monitors will be in charge of supervising both the input data being consumed by an AI and its predictions in view of safety-relevant consequences as well as deviations from physical-mathematical knowledge. Moreover, these monitors are intended for usage during an AI's development cycle and also after its deployment. Throughout this section, we will simply refer to these cause-effect-aware conformity-checking units as *AI-monitors* for short. These monitors are not intended for real-time supervision of an AI-component first and foremost, although it is entirely possible.

The remainder of this section is structured as follows:
First, we describe our considered (sub) use cases (paragraph 4.9.2), before we elaborate on the relevant knowledge (paragraph 4.9.3) and its formal representation in terms of SCMs (paragraph 4.9.4). Finally, we put everything together and conclude with our proposed knowledge conformity concept of AI-monitors (paragraphs 4.9.5 - 4.9.5.5), before we end with a short summary of our knowledge conformity concept (paragraph 4.9.5.6).

4.9.2 Use Cases

In the course of this project's lifespan, we will test and evaluate all our proposed concepts on Use Case 2 (Complex Lane Change). In particular, we will focus on its following sub use cases (SUC):

- **SUC-2.4**: *Lane Change in Multi-Lane Road*
- **SUC-2.11**: *Lane Change Prediction of Exo-Vehicles in Multi-Lane Road*

Concretely, we assume two different AI-systems, each specialized in solving a single SUC:

- In **SUC-2.4**, an AI *controls* an ego-vehicle in such a way that it is able to weave its own way through a multi-lane road traffic by performing a lane change on its own and in such a way that it neither results in an accident with other traffic participants nor in undercutting safety-margins.

- In **SUC-2.11**, an AI *predicts* another traffic participant's intention for a lane change in the near future. For this purpose it is assumed that the AI makes its prediction conditioned on the consumption of spatiotemporal information on the past trajectories of all surrounding traffic participants. The AI-system shall then assign probabilities to all vehicle drivers in a scene for whether they will perform a *Lane Change Left*, *Lane Change Right* or *No Lane Change*.

4.9.3 Knowledge Source(s)

Now, we turn our attention to the knowledge sources, that we intend to use in this project. Although there exists a vast amount of qualitatively different knowledge sources, we decided to restrict ourselves to those that are directly linked to physical-mathematical knowledge and those best suited for application in SUC-2.4 and SUC-2.11. Specifically, we use a *Vehicle Dynamics Model* [61, 62]. Knowledge about the way vehicles physically move in the first place as a function of their control inputs is encoded in this model. Technically, the vehicle's current state (s_t) consists of its actual position and velocity, while its control inputs are given by the steering angle, throttle and brake control. The latter can be converted into longitudinal and lateral accelerations. In addition, a realistic and carefully chosen vehicle dynamics model can not only enforce constraints between two consecutive states (s_t, s_{t+1}) in a natural way, but it can also enforce constraints within a single state, e.g., between position and velocity like arising from the fact that lateral motion is always accompanied by longitudinal motion. Moreover, it can likewise render certain values for acceleration and deceleration, respectively, unrealistic and help rule them out from the very start.

4.9.4 Knowledge Representation (Concept)

As pointed out in the introduction (paragraph 4.9.1), we aim at representing our vehicle dynamics model in terms of an SCM due to its close connection to *Causal Reasoning*. By definition, Causal Reasoning [60] is the process of drawing conclusions from a causal model, similar to the way probability theory reasons about the outcomes of random experiments. However, since causal models are thought of as knowledge of the data-generating process[1], they contain more information than probabilistic models and are thus more powerful, as they allow to analyze the causal effects entailed by model modifications or changes in prior distributions [63]. Questions posed to such a causal model are phrased as *Causal Queries*. They are at the heart of Causal Reasoning and can be differentiated into *three levels* of increasing complexity of the *Causal Hierarchy* [60] as depicted in Fig. 31:

Level (Symbol)	Typical Activity	Typical Questions	Examples
1. Association $P(y\|x)$	Seeing	What is? How would seeing X change my belief in Y?	What does a symptom tell me about a disease? What does a survey tell us about the election results?
2. Intervention $P(y\|do(x), z)$	Doing, Intervening	What if? What if I do X?	What if I take aspirin, will my headache be cured? What if we ban cigarettes?
3. Counterfactuals $P(y_x\|x', y')$	Imagining, Retrospection	Why? Was it X that caused Y? What if I had acted differently?	Was it the aspirin that stopped my headache? Would Kennedy be alive had Oswald not shot him? What if I had not been smoking the past two years?

Fig. 31: The *Causal Hierarchy*. Causal Query at level i ($i \in \{1, 2, 3\}$) can only be answered, if information from level i or higher is available [64].

Technically, SCMs are a functional-graphical representation of knowledge, since they make explicit use of a graphical representation in terms of a *Directed Acyclic Graph* (DAG) whose directed edges are associated with a set of functions (see Fig. 32). The key characteristic of SCMs is that they represent each variable as a deterministic function (f_X and f_Y in Fig. 32) of its direct causes optionally including latent exogenous noise variables (ϵ_X, ϵ_Y and U in Fig. 32). The latter stand for causes lying outside the SCM that are not explicitly modeled. If we wanted to have a probabilistic causal model, the exogenous noise variables would be drawn from a (joint) probability distribution effectively defining a joint probability distribution over the endogenous variables (X and Y in Fig. 32). Due to the close connection between SCMs and causality, they are best suited for answering causal queries in general and in a mathematically sound and transparent way. This is one of the reasons, why

1. e.g., the laws of physics encoded in our vehicle dynamics model

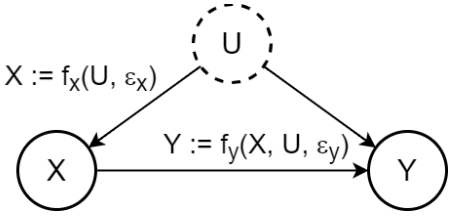


Fig. 32: An SCM consists of both a set of causal mechanisms (f_X and f_Y) and a DAG modeling the flow of causation between variables (directed edges). The shown SCM is composed of a cause (X) and its direct effect (Y), both confounded by a latent variable (U). Nodes associated with the exogenous noise variables ϵ_X and ϵ_Y are omitted for the sake of clarity.

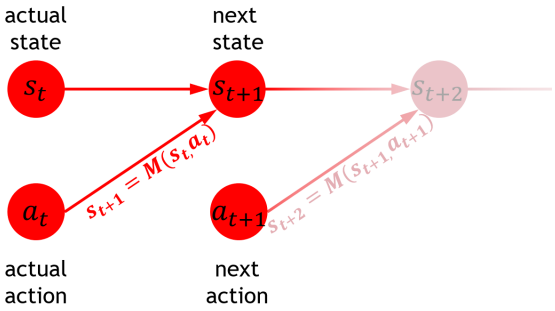


Fig. 33: A high-level version of our Vehicle-SCM and its application in the process of vehicle trajectory generation. The vehicle's current state-action-pair (s_t, a_t) is mapped via a functional causal mechanism, the vehicle dynamics model $M(s_t, a_t)$, onto the vehicle's next state s_{t+1}, while obeying the laws of physics encoded in $M(s_t, a_t)$. Subsequent vehicle states are obtained by sequential application of the vehicle dynamics model.

they have already found widespread application in a variety of other fields like e.g. epidemiology [65] and social science [66] and why we decided to represent our knowledge of vehicle dynamics using this framework.

In the following, we refer to the SCM-representation of our vehicle dynamics model as the *Vehicle-SCM*. A high-level version of such a Vehicle-SCM is depicted in Fig. 33. There, individual vehicle-state-related variables are subsumed in the s_t variable. The same argument applies to various, independent action variables a_t. In the Vehicle-SCM, the current state and action are modeled as parental nodes both pointing in yet another node representing the vehicle's next state, as the latter causally emerges from the former ones. The way the next state's value s_{t+1} results from the actual values of its parental nodes (s_t, a_t) is specified by a function and is distinctive of the chosen vehicle dynamics model M. Interestingly, vehicle trajectories can be conceived of as being generated by repetitive application of the Vehicle-SCM as indicated in Fig. 33.

4.9.5 Conformity Check (Concept)

As mentioned in the introduction, the main objective of our knowledge conformity concept is to derive answers to causal queries about an AI model's input data and predictions by taking physical-mathematical knowledge explicitly into account. Therefore, our knowledge conformity concept views the AI as a component of an environment. In our case, the environment consists both of a multi-lane road and of multiple interacting vehicles, which could also occlude each other temporarily. Therefore, the environment is naturally modeled as a POMDP[2], which is convertible into an SCM. With that said, our concept basically decomposes into two elements that are subject of the following paragraphs:

1) Preparation of a realistic Vehicle-SCM
2) Causal-Query-Answering AI-Monitors

4.9.5.1 **Preparation of a realistic Vehicle-SCM**: Before being able to resort to some kind of model for the purpose of answering causal queries, one needs to design and validate such a model first and foremost. Since our AI-monitors will make use of the Vehicle-SCM later on, we would like to have a lightweight, but nonetheless reliable vehicle dynamics model in the sense that it makes decently accurate predictions. For these reasons, we consider a *Circle Model* in the first place. It propagates a vehicle along a circular segment, effectively ensur-

2. POMDP: Partially Observable Markov Decision Process

ing coupled longitudinal and lateral motion. In the course of this, the radius of the imaginary circle is derived from the vehicle's state and action. As the Vehicle-SCM is the backbone of our knowledge conformity concept, it has to be validated by means of real-world trajectories (as insinuated by the earth-like pictogram in Fig. 30). This is necessary, because the results provided by the AI-monitors are only as good as the model's predictions. Therefore, we will continuously refine and validate our Vehicle-SCM by means of real-world vehicle trajectories. In the course of this, we will assess its performance by reference to its capability to reconstruct trajectories. The *highD*- (https://www.highd-dataset.com) and *AUTOMATUM*-drone [67] datasets (https://www.automatum-data.com) prove themselves useful for this purpose and preliminary experiments look promising.

4.9.5.2 **Causal-Query-Answering AI-Monitors**: Remembering that causal queries are associated with three, conceptually different levels of the Causal Hierarchy, we will discuss each level one by one in the remaining paragraphs.

4.9.5.3 **Associational Queries**: At the *associational* level of the Causal Hierarchy, the AI-monitor is intended to constantly check whether trajectories of other traffic participants comply with our Vehicle-SCM, as the trajectories are consumed by an AI for its decision making process. If the model was not able to reconstruct the trajectories, as perceived e.g. by an AI, using a conveniently chosen action sequence, they would be considered flawed or at least unreliable. In this case, the AI-monitor could issue a warning about the discrepancy between perceived and reconstructed trajectory either to the driver or raise an exception that is caught and processed by other components. With this said, we are interested in answering associational queries as exemplified in the following example (cf. SUC-2.4 & SUC-2.11).

Example of an Associational Query:

"*What is the most likely action sequence that conforms with our vehicle dynamics model, given I perceive a certain trajectory?*"

expressed more formally:

$$a^{*(H)} = \underset{a^{(H)}}{argmax}\ P_M(a^{(H)}|\tau^{(H+1)}) \tag{6}$$

where $a^{(H)} = (a_0, \ldots, a_{H-1})$ is the action sequence of length H, P_M stands for the probability under the model assumptions M, $\tau^{(H+1)} = (o_0, o_1, \ldots, o_H)$ stands for the perceived trajectory of length $H+1$ and where o_t indicates individual observations of the trajectory in case the real vehicle state s_t is unobserved. The latter can easily happen when vehicles occlude each other temporarily. In this case, the real (currently unobserved) state s_t emits an observation o_t usually carrying less information than the full state s_t and taking account of the transient occlusion.

Another imaginable sanity-check relying on pure associations could take infrastructural information except the pure trajectories into account. For instance, one could check, whether trajectories consistently range between the multi-lane road boundaries at each point in time. If this was not the case, either the perception might have gone wrong, a potentially hazardous event occurred or the model assumptions are incomplete in the sense that the model is insufficient for coping with this situation. In either case, extra measures should be taken then.

4.9.5.4 **Interventional Queries**: At the *interventional* level of the Causal Hierarchy, our AI-monitors are intended to answer causal queries as outlined in the next example (cf. SUC-2.11).

Example of an Interventional Query:

"*Would my AI-component's belief in another driver's intention for a lane change persist, if I actively changed the AI-component's consumed trajectories a bit?*"

expressed more formally:

$$l^* = \underset{L}{argmax}\ P\left(L\ \middle|do(\tau_1^{(H_1+1)}), \ldots, do(\tau_V^{(H_V+1)})\right) \tag{7}$$

where P signifies the AI-component predicting upcoming lane changes of exo-vehicles (cf. SUC-2.11), l^* indicates the most probable lane change label among the set of possible lane changes L,

$\tau_v^{(H_v+1)}$ indicates vehicle-specific trajectories of varying length and *do* represents the *do*-operator responsible for the active manipulation of trajectories irrespective of their natural formation process without active intervention [60]. Obviously, a robustness or sensitivity analysis, respectively, can be conceived of as an interventional query posed to an AI-component and based on certain model assumptions. In this way, one could figure out, if the AI-prediction (cf. SUC-2.11) was reliable or would change, if the input trajectories were actively modified.

Moreover, in view of SUC-2.4, interventional queries can also assess whether harmful consequences would be entailed by following certain trajectories, like e.g. accidents or the risk of undercutting safety-margins. This is so, because trajectories are the direct effect of sequentially applying a certain action sequence starting from an initial state. Therefore, having a validated model of the vehicle dynamics allows one to act out different action sequences in an imaginary space, to evaluate their effects with regard to safety goals and everything without the need of interacting with the environment directly.

4.9.5.5 **Counterfactual Queries**: Answering *counterfactual* queries is seemingly intractable from a computational point of view, because evaluating counterfactual queries usually involves the derivation of posterior distributions given some evidence or hindsight data. An exact evaluation is often beyond reach, since it entails a usually intractable evaluation of the probability for the evidence that involves an integration over the latent state space. In any case, an exact derivation of the posterior distribution is often unnecessary and, fortunately, recent approaches have been devised that are based on Deep Learning techniques promising to overcome this problem via approximations while yielding satisfactory results [68]. Anyway, in our case, the hindsight data could be given in terms of vehicle trajectories (made up of a sequence of observations o_t) that were recorded by following a certain policy, as e.g. in SUC-2.4. There, one could be interested in answering the following counterfactual query.

Example of a Counterfactual Query:

"Would the AI also have maintained the safety margin to the front car, if this had performed an emergency braking all of a sudden, although it actually did not?"

expressed more formally:

$$x^{\pi *} = \underset{x^{\pi}}{argmax}\ P_M(x^{\pi}_{\tau^f} | \tilde{\tau}^f, \tilde{x}^{\pi}) \tag{8}$$

where again P_M stands for the probability while adhering to the model assumptions M, τ^f and $\tilde{\tau}^f$ are the hypothetical emergency-braking-related and true trajectory of the front vehicle, respectively, x^{π} and $\tilde{x}^{\pi}$ indicate the hypothetical and the actual safety margins attainable by following policy π, respectively, and the τ-subscript of x_τ in $P_M(x_\tau)$ is a shorthand notation of $P_M(x|do(\tau))$.

Evaluation of counterfactual expressions like the one given above will enable our AI-monitors to actually unveil, whether an AI's policy π was error-prone in a hypothetical and often unforeseeable emergency braking situation or whether it were even capable of safely coping with it. It does so by comparing the true safety margin (the evidence: $\tilde{x}^{\pi}_{\tau}$) to another, hypothetical safety margin (the one that could have been: $x^{\pi}_{\tau^f}$), if the front vehicle had performed an emergency braking maneuver. The comparison is made by ensuring that the AI both sticks to the very same policy π and starts from the same initial state s_t. A similar procedure, which compares the performance of two distinct policies, however, was utilized for policy-improvement in [69].

Having assessed an AI's performance in hypothetical emergency braking situations could not only shed light on an AI's policy-related (π) insufficiencies, but could rather serve as new knowledge that could be integrated into the AI for safety-oriented improvement of its performance later on. In this sense, counterfactual-based AI-monitors could effectively reveal scenario-specific and safety-jeopardizing causal impact factors.

4.9.5.6 **Summary**: In summary, the main objective of our knowledge conformity concept is to provide answers to causal queries along all levels of the Causal Hierarchy, e.g., about the trajectories that are consumed by an AI for the purpose of its

decision making process. Answers to these causal queries can shed light on the physical validity of perceived trajectories (*associations*), on the sensitivity of an AI-component's response when the trajectory changes (*interventions*) and on the causes of an AI-component's potential malfunction (*counterfactuals*). In order to do so, having a model of the trajectory generation process is inevitable. Therefore, a trajectory is viewed as the sequential application of an action sequence to successively available states. The model behind the trajectory-generation process itself is translated into the SCM framework. It is also capable of taking account of transiently unobserved vehicle states, as they often occur by mutual vehicle occlusion. Additionally, physical-mathematical knowledge is explicitly considered in terms of a Vehicle-SCM and integrated as a single component into the trajectory generation model.

4.10 fortiss GmbH

Topic overview:

Use Cases (Scenario)
UC 2

Knowledge Source
Traffic rules and geometric relationships between different road users or between road users and the environment

Knowledge Formalization & Representation
Equations and inequalities

Conformity Check
Automated framework which tests if Neural Networks (that control autonomous cars in a simulated environment) obey traffic rules

4.10.1 Introduction

A common approach to autonomous navigation is Imitation Learning (IL), where the goal is to learn a policy π that imitates the behavior of an expert π^*. In our setup, the policy is a mapping from observation x and high-level command c, such as "follow the lane", "turn left", and "turn right", to a trajectory $\tau = \pi(x, c)$, which is provided to a low-level controller to output actions. For training, an expert policy is rolled out in the environment to collect a dataset $\mathcal{D} = \{(x_t, c_t, \tau_t)\}_{t=1}^{T}$.

However, simply minimizing the distance between the predicted trajectory and a rolled out expert policy is insufficient for dealing with complex driving scenarios. Generally in each situation there exists a distribution of valid trajectories which depends on many factors including traffic rules, lane width and the intentions of other road users. It is difficult to infer the causal relationship from a single example of this distribution. For better generalization to novel situations we want to integrate the knowledge about traffic rules and the dependency of the set of valid trajectories on geometrical relationships into the learned policy.

4.10.2 Use Cases

We validate our approach based on the use case of complex lane change scenarios. However, we do not focus on a specific sub-use case.

4.10.3 Knowledge sources

We consider two different sources of knowledge: Traffic rules and geometric relationships between different road users or between road users and the environment.

All participants of traffic have to obey certain traffic rules. These rules differ from country to country and can even change over time (e.g. if there is new legislation). This means that there should be an easy way to integrate them into autonomous cars, in order to be able to react to new laws or regulations. Traffic rules are not only important for behavior generation (of the autonomous car itself) but also for forecasting of the behavior of other traffic participants.

An example of a geometric relationship is the distance between the ego vehicle and the vehicle ahead. While driving, we constantly check the distance and adjust our own behavior accordingly to avoid accidents. We intend to integrate this knowledge by explicitly modeling such relationships in the trained policy. Learning to control the distance to other road users is an additional learning signal that can be derived even without a target trajectory given by an expert and thus can be used even in the presence of strong data augmentation, e.g., when the behavior of other road users is substituted. In this case, the specified target trajectory may no longer be valid, but the safety distances should still be adhered to.

Another example is the lateral distance to the center of the lane. Although vehicles typically drive in the center, a policy that learns to associate predicted trajectories with this offset might be more robust to changes of other factors.

4.10.4 Representation/Formalization concept

Based on the given sequential training dataset $\mathcal{D}$ we derive the geometric relationships of interest for each timestep t and store them in auxiliary variable u_t. The relationships may refer to a future point in time or time period, i.e. $u_t = f(x_{t+\Delta t})$ or $u_t = f(x_{t:t+\Delta t})$. If the geometric relationships cannot be extracted from the raw sensor data with the required precision, an HD map may be necessary. For easier computation we assume that the raw sensor readings have already been processed into a compact representation that contains information about the environment and the states of other road users. How the relationships are encoded and processed and which ones are most appropriate (e.g.

maximum or average distance to the center of the lane) remains to be explored.

The additional variable constrains the set of valid trajectories and plays a similar role as the high-level command, but on a more fine-grained level. During training, the model $\pi(x, c, u)$ learns to output trajectories that meet the specifications contained in u, in addition to the usual requirements such as driving towards the destination and obeying traffic rules.

4.10.5 Checking for Knowledge-Conformance

We plan to develop a "Knowledge-Conformance-Framework" as shown in Fig. 34. The framework will test autonomous vehicles (in a simulation) and check if the obey traffic rules. As a simulator we will use BARK[70], which has been developed at fortiss. The traffic rules are modeled using mathematical/logical formulas called ”Linear temporal logic" (LTL)[71]. These formulas are applied to the state of the autonomous vehicle in the simulation and allow for a definite decision if traffic rules have been breached.

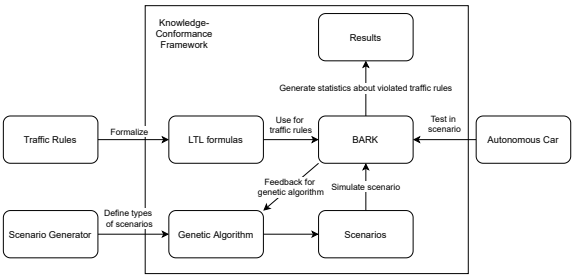


Fig. 34: This graphs show the "Knowledge-Conformance-Framework.

The framework will generate a diverse set of scenarios where the autonomous vehicle is tested. It will be possible to compare different autonomous agents, e.g. one without added knowledge vs. one where the knowledge has been inserted. The results of these comparisons will be analyzed (e.g. through statistics). Additionally, we plan to make use of genetic algorithms to find very challenging scenarios where traffic rules are often breached. This will allow to optimize the autonomous vehicle in these difficult situations.

4.11 Fraunhofer-Institut für offene Kommunikationssyteme (FOKUS)

Topic overview:

Use Cases (Scenario)
UC 3

Knowledge Source
Traffic Regulations / Legal Knowledge

Knowledge Formalization & Representation
Defeasible Deontic Logic / LegalRuleML

Conformity Check
Logical Consistency & semantic Correctness

4.11.1 Introduction

Formalizing knowledge usually requires the person who formalizes to be an expert in both the target domain (which has to be formalized) and the domain of formalization/logics in order to choose a suitable formalization framework (logic) and to map the (natural language) knowledge on this very formal framework. Even if the person was a domain expert in both fields, it would require a tedious amount of work of manual labor without a proper tool supporting this process, even more so if the final representation should be machine-readable. Our goal is to remedy some of these issues by providing a domain expert (in this case a legal expert) with a software tool that removes most of the requirement about logic and formalization skills. Therefore we propose the Legal Norm Editor (LNE), providing an easy-to-use interface for domain experts who are not logicians to formalize legal knowledge and validate the result.

4.11.2 Knowledge sources

In our scenario, the relevant knowledge encompasses legal texts i.e. the traffic law (Straßenverkehrsordnung/StVO), court cases and other regulations. Moreover, a description of traffic scenes and traffic behavior is required that interacts with legal knowledge (e.g. a traffic participant in the StVO should be the same as a traffic participant in the traffic scene) in order to create applications such as a legal analysis of behavior in traffic situations.

4.11.3 Representation/Formalization concept

Traffic scenes and their representation. The description of a traffic scene will start with a schema of things (e.g. the concept of a vehicle) that might occur in such situations, their attributes (e.g. cars possess a velocity vector) and relational aspects between the entities or attributes (e.g. one car could be behind some other car). Such schema is called the TBox (T stands for terminology) of an ontology. The actual objects that represent existing things are part of the ABox (A stands for assertion) which instantiates the schema of an ontology. An example how a part of such an ontology could look like is illustrated in figure 35. We aim at creating such ontology during this project which will be capable of describing all aspects of a traffic scene relevant to our applications in this project. For all applications we formulate *competency questions* which are informal questions the model (the ontology) should be able to answer e.g. *"Is vehicle A on the same lane as vehicle B?"*. Those questions help us in two ways with the development of the ontology: It guides the creation of concepts and relations since both are parts of competency questions and provides us with the means to assess the adequateness of our ontology.

Legal norms and their representation. According to [72] legal norms can be understood as normative conditionals which are represented as $if\ A_1, ..., A_n\ then\ C$. $A_1, ..., A_n$ represents the pre-conditions of a rule and C the normative effect. In contrast to material implications used in e.g. propositional logic and programming languages, normative conditionals are generally defeasible. A defeasible logic allows for non-monotonic reasoning. A conclusion of a defeasible conditional is inferred only if all pre-conditions hold and there is no evidence to the opposite conclusion. This mechanism allows for elegant formalization of legal norms as general principles and for expressing exceptions to these principles at the same time.

Legal norms can have different purposes. They can express facts (e.g. a contract that defines property ownership of a certain asset). These are referred to as *constitutive norms*. Another type of legal norms has a regulative function and is known as *prescriptive norms*. *Prescriptive norms* are substantially different from *constitutive norms* in that they do not deal with facts which can be true or false but instead express obligations, permissions and related concepts. The philosophical discipline

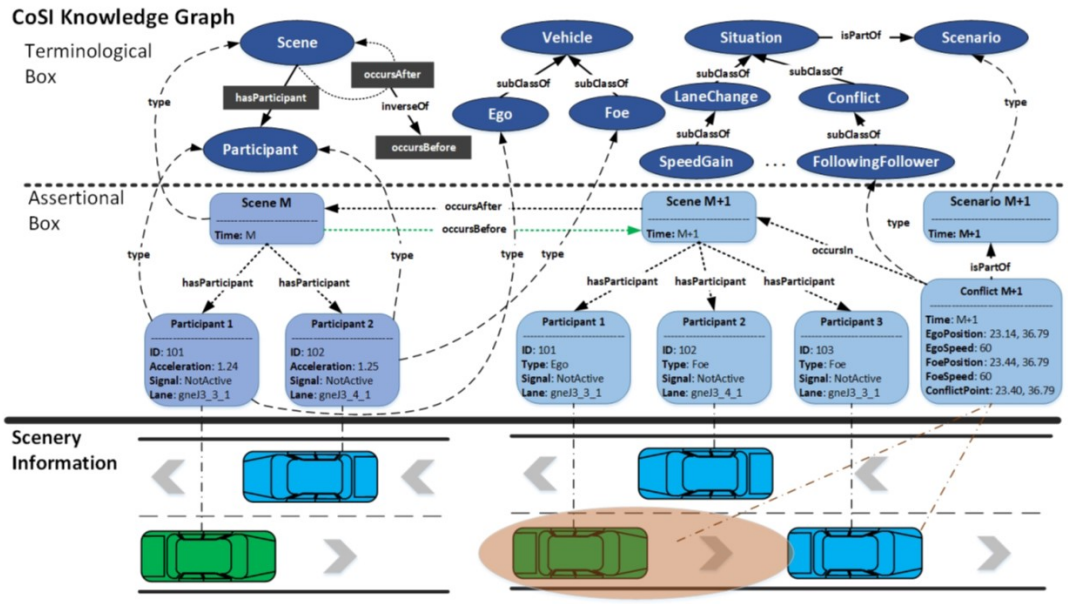


Fig. 35: TBox and ABox in a traffic scene ontology snippet.

which is concerned with this concepts is known as *deontic logic*. There are different ways to model deontic statements. A common approach chooses to implement *deontic logic* as a variant of *modal logic* introducing the following deontic operators (see [73] for more details):

- $[OBL]p \equiv \neg[PERM]\neg p$: if p is obligatory, then its opposite, $\neg p$, is not permitted.
- $[FOR]p \equiv [OBL]\neg p$: if p is forbidden, then its opposite is obligatory.
- $[PERM]p \equiv \neg[OBL]p \wedge \neg[FOR]p$: if p is permitted, p is neither obligatory nor forbidden.

In current research different logic formalisms are discussed for formalization of legal norms, like *Defeasible Deontic Logic* or most recently *Reified Input/Output Logic*. However, there is still no consensus on which logic framework is best suited. Therefore, instead of committing to a specific logic formalism, we choose to represent the legal norms in LegalRuleML ([73]), which is an machine readable interchange format based on XML. LegalRuleML is developed specifically for the needs of the legal domain, supporting besides constitutive statements also prescriptive statements by introducing deontic operators. It also features different mechanisms to express defeasibility of rules. Furthermore, LegalRuleML supports a close connection between formalized rules and the actual sections of the legal texts, which a rule is representing. Since LegalRuleML origins from the Semantic Web community it natively supports referencing knowledge in a linked data fashion. Finally, LegalRuleML supports authorship tracking, different roles and figures and has a notion of temporality of and in rules and jurisdiction. For a in-depth description of LegalRuleML the reader is referred to [73].

However, LegalRuleML is only an interchange format for legal rules and has no standard semantics. Therefore, it is not possible to perform reasoning tasks directly on a LegalRuleML knowledge base. First, the rules stored in the LegalRuleML format must be mapped onto a specific normative logic representation on which the reasoning task can be performed. Such a mapping is unique for each target logic formalism. Lam, Hashmi, and Scofield proposed in [74] a mapping from LegalRuleML to a *modal defeasible logic* and used the SPINdle reasoner

(see [75]) for the reasoning step.

Reasoning. In our work we want to go a step further by (1) incorporating additional knowledge for the reasoning process and (2) evaluate a new kind of learnable neuro-symbolic reasoning approach. As said before, LegalRuleML is a fruit of the Semantic Web community with the ideal of an network of inter-linked knowledge atoms distributed across the entire internet. Hence, LegalRuleML is designed to reference rules, operators and even single variables to specific concepts modeled through OWL-ontologies. This additional knowledge can then be utilized during the mapping process from LegalRuleML to the normative target logic description in order to e.g. introduce a certain semantic to a rule or to apply rules, that are formulated only for objects of a certain class also to all sub-class entities. E.g. for a computer it is not quite obvious that regulations that apply to traffic participants should automatically also apply for pedestrians, bicyclists and car drivers in the same time, since these are also traffic participants. With the application of domain knowledge represented in an ontology during the mapping step this problems turns to be an easy task. The choice of a normative target logic depends on the supported logic formalisms of the favored reasoning engine.

Besides the well studied classical reasoning engines, there are concepts recently proposed in academia to combine symbolic AI (like logic and ontologies) with sub-symbolic AI methods (neural networks) in order to build neuro-symbolic reasoning engines. While classical reasoners use an algorithm that encodes a reasoning strategy (like the DPLL algorithm), neuro-symbolic approaches aim to learn the reasoning strategy and steps using machine learning techniques. A promising approach for neuro-symbolic reasoning called Logic Tensor Networks (LTNs) is presented in [76]. The authors show that LTNs are able to use full first-order logic with function symbols by embedding these logic symbols into real-valued tensors. They propose a neural-symbolic formalism called Real Logic in addition to the computational model that is designed for defining logical expressions suited for tensorisation in LTNs. Real Logic is a many-valued, end-to-end differentiable first-order logic. It consists of sets of constant, functional, relational and variable symbols. Formulas build from these symbols can be partially true and therefore Real Logic includes fuzzy semantics. Constants, functions and predicates can also be of different types represented by domain symbols. The logic also includes connectives $\diamond \in \{\neg\}, \circ \in \{\wedge, \vee, \rightarrow, \leftrightarrow\}$ and quantifiers $Q \in \{\forall, \exists\}$. Semantically, Real Logic interprets every constant, variable and term as a tensor of real values and every function and predicate as real function or tensor operation. Therefore, LTNs are able to compute an approximate satisfiability by mapping logical expressions to real-valued tensors.

To use LTNs as a neuro-symbolic reasoner (or classical reasoning engines) we propose the following pipeline: (1) Encoding of legal norms in LegalRuleML and modeling domain and world knowledge with ontologies, which are referenced by the formalized norms. (2) Map the two knowledge bases into a single first-order-logic representation. (3) Convert that representation into a Real Logic representation and (4) training/reasoning with LTNs or classic reasoners.

Knowledge crafting. Creating and validating a logic representation of a medium-sized (legal) corpus is quite time-consuming. The formalization of the European GDPR took four months by an expert on formalization without validation [77]. Since we aim at providing domain experts with the means to formalize knowledge themselves we need a more sophisticated methodology and software support for this process. Therefore our goal is to develop a formalization tool a non-logician can employ to formalize legal knowledge.

This software takes legal text in Akoma Ntoso / LegalDocML[78] as input. In this format, the text is already structured by articles and paragraphs. The LNE will feature an annotation tool in which the user can map parts of the text to entities of ontologies. This encompasses mapping text expressions to

- single entities, attributes or relations of the traffic scene ontology,
- new entities the user will create himself and build a simple terminology ontology from this, and,
- normative structures.

Normative structures are constitutive or prescriptive norms which will be accessible

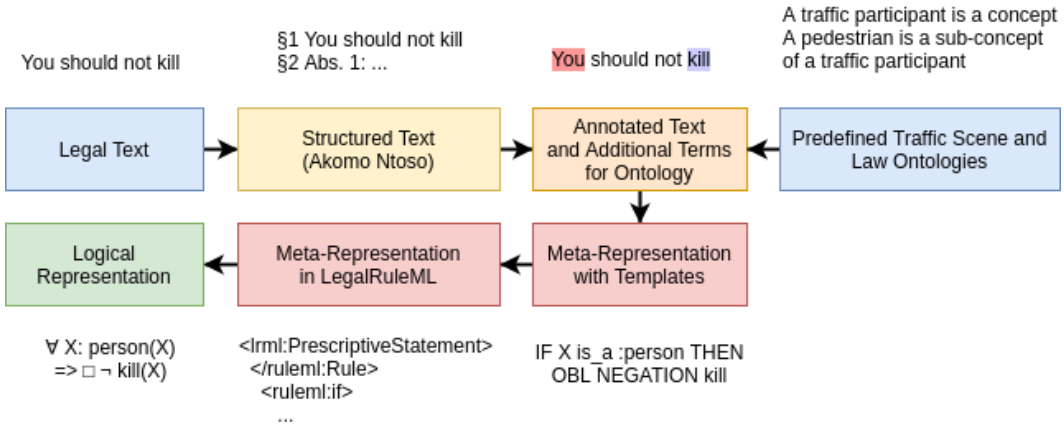


Fig. 36: Knowledge representation layers in the LNE.

for the user as templates, e.g.

If an ambulance is approaching, make room.

could be matched by a template like this:

IF *an ambulance is approaching* THEN OBLIGATION *make room*

Those templates could also be nested when formalizing more complex statements. The resulting semantically structured representation together with the ontologies will be encoded in LegalRuleML and subsequently in a suitable target logic that can deal with normative contexts and has yet to be determined. Together with queries described in the next paragraph on conformity checks that aid in the validation of the formalization, this representation will serve as input for the reasoning pipeline above yielding the results of the checks.

4.11.4 Conformity check

To assess the consistency and correctness of the set of formalized traffic rules we are planning to apply multiple testing techniques, which cover (1) the logical consistency, (2) the semantic correctness and (3) the check for undesired behavior emerging from the increasing complexity of the rule set and unforeseen eventualities.

1) **Automated checking of the previously formalized rules for logical consistency:** During the formalization process the consistency of the corpus of already formalized rules can be continuously assessed. A corpus of logic statements is consistent if no contradiction can be derived from it. If that is not the case, then the corpus is inconsistent and needs to be revised.
2) **Test-driven formalization of traffic rules:** We implement test cases (hypotheses) of undesired behavior (e.g. crashing into persons). A model finder checks if at least one of the hypotheses can be fulfilled under the given set of formalized rules. If that is the case, this might be a hint, that the formalization of traffic rules must be revised.
3) **Conformance checks via simulations of scenarios related to Use Case 3:** First, test scenarios based on Use Case 3 are developed in a simulation environment (e.g. CARLA). The autonomous vehicle must behave in the desired manner within these scenarios, i.e., driving in accordance with the rules or, in special cases, allowing controlled rule exceptions. The formalized traffic regulations of the StVO form the basis for this.

4.12 Fraunhofer-Institut für Intelligente Analyse- und Informationssyteme (IAIS)

Topic overview:

Use Cases (Scenario)
UC 1

Knowledge Source
Street Maps, HD Maps

Knowledge Formalization & Representation
Knowledge Graph + Visual Representation

Conformity Check
Conformity Metrics from Prediction and Knowledge Overlay

4.12.1 Introduction

Our approach is based on the street-map based validation algorithm for semantic segmentation in autonomous driving, as presented in [25].

We develop a conformity check method that validates predictions from AI models by using prior knowledge. This is inspired by the idea of informed machine learning [2, 3]. It describes the concept of utilizing prior knowledge in data-driven methods, for example to ensure knowledge conformity. Given the context of autonomous vehicles, knowledge conformity can be crucial aspect to ensure robustness and safety.

We focus on the task of environmental perception, which is important for autonomous vehicles in order to assess the surrounding traffic scene and understand its context [79, 80]. State-of-the-art methods and architectures of deep neural networks already allow to learn models that perform impressively well. However, it can still be observed that certain regions or objects are not detected correctly. This could be due to unusual environment conditions or to difficult traffic scenes with object occlusion, which complicates their correct detection. As an example, roads and pedestrian walks could be mixed up in difficult lighting conditions or unusual terrain, such as in the prediction image in the left from Figure 37, or pedestrian could not be detected because they are occluded by parking cars.

In our approach, we propose to utilize spatial prior knowledge given in street maps to check the knowledge conformity of AI models for perception in autonomous driving. As illustrated in Figure 37, we suggest to compare detected objects or segmentation masks to the structured semantic information

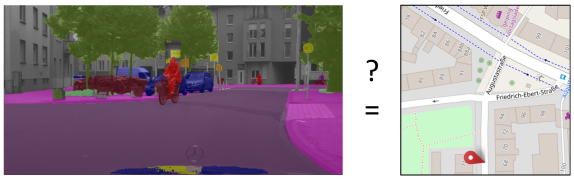


Fig. 37: Idea of street-map based validation, as proposed in [25]. The left image shows a segmentation of a traffic scene in the Cityscapes dataset [81] and the right image shows the corresponding map from OpenStreetMap [24]. In our approach, we employ such street maps to check the knowledge conformity of AI models.

in street maps. In particular, we present a method to compute an overlay of both prediction and knowledge in order to quantify their conformity.

4.12.2 Use Cases

We develop our algorithm for use case 1 of the KI-Wissen project, i.e., pedestrian detection under occlusion. We consider traffic scenes in city environments, such as in the CityScapes and CityPersons dataset [81, 82]. This is relevant because of the challenge of multiple objects and potentially crowded scenes, which can lead to occlusion of individual traffic participants.

4.12.3 Knowledge sources

Our source for the conformity check algorithm is geometrical knowledge given in street maps. Street maps are available for nearly all places world wide. Prominent examples are OpenStreetMap [24] or GoogleMaps. These maps contain information about static and semantic regions of the traffic scene, e.g. roads, pedestrian walks, buildings, etc. HD maps constitute an even more sophisticated knowledge source and contain more precise and fine-grained information.

4.12.4 Formalization/Representation concept

The knowledge from street maps can be represented in a graph. Especially for OpenStreetMap the road network topology is given by a graph, that consists of nodes, which represent positions of intersections, and edges, which represent the connecting roads.

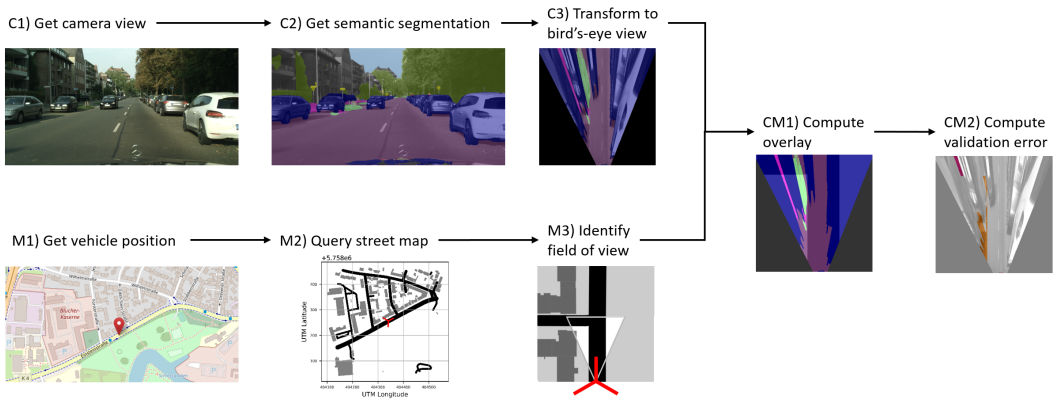


Fig. 38: Approach of street-map based validation, as proposed in [25].

The edges in this graph also contain further information about the road type, e.g. highway or living street, which adds valuable semantic context.

For our conformity check algorithm, we transform the graph into a visual representation.

4.12.5 Conformity check

Our approach is to check the conformity of pedestrian detection models with geometrical knowledge from street maps. For this, we follow the street-map based validation algorithm, as proposed in [25] and illustrated in Figure 38, and adapt it to our use case for pedestrian detection. In the following we shortly describe each step within the approach.

Step C1) Get camera view: We process images from the vehicle's front view of a traffic scene. The illustration shows an example from the CityScapes [81] dataset. In order to apply the approach to pedestrian detection, we extend the dataset to CityPersons [82].

Step C2) Get semantic segmentation: This step stands for the application of an AI model. In the approach image, a neural network for semantic segmentation is used to predict pixel-wise classes, here using the ERFNet architecture [83]. The segments are visualized using the standard color map: *road* is violet, *car* is blue, *vegetation* is green, *person* would be red, etc. For additionally retrieving pedestrian bounding boxes, an object detection model, e.g. Faster R-CNN [59], should be applied here.

Step M1) Get vehicle position: We get the position by reading the GPS coordinates of the vehicle and thus retrieve latitude, longitude and the heading. Since the GPS coordinates do not necessarily provide the exact position, the usage of localization correction can be useful. For further details, please see [25].

Step M2) Query street-map graph: Based on the given vehicle position, we query the street map graph for the surrounding area. As described above, this graph contains a street network, where the roads are given as connected line segments.

Step C3) Transform to bird's-eye view: The next step is to prepare the camera image, or rather the neural network's prediction based on that image, e.g. the pedestrian bounding boxes, for an overlay with the prior knowledge from the street map. Therefore, we transform the image into a bird's-eye view, which corresponds to the view space of the street map using an image-based perspective transformation.

Step M3) Identify field of view: Similarly, the street map graph needs to be prepared for the overlay. For this, we first transform it into a visual representation by plotting the semantic map elements in an image. As shown in Figure 38, the road is plotted with black lines and different types are taken into account to depict the corresponding road width. Buildings are also plotted, as shown by the gray polygons. The image is then rotated and zoomed so that it corresponds to the potential field

of view from the camera mounted on the car.

Step CM1) Compute overlay: In this central step, the neural network prediction and the prior knowledge from the street map are combined in an overlay. In Figure 38, the validation of semantic road prediction is illustrated by showing the road as a transparent black area on top of the colored semantic segmentation. For the use case of pedestrian detection, the positions of the pedestrians should also be included in the overlay.

Step CM2) Compute validation error: Finally, we can evaluate the knowledge conformity metric. For this, we need to analyze the relation between the pedestrian positions and the street map elements. For a quantification of the knowledge conformity, additional rules might be useful. For example, a pedestrian can be positioned on a road, but is generally not located on a tree or a car. However, if such unusual cases would be the output of the neural network, our conformity check algorithm could help to identify this and raise a potential warning.

4.13 FZI Forschungszentrum Informatik (MPS)

Topic overview:

Use Cases (Scenario)
UC 2, UC 3

Knowledge Source
Traffic rules, social norms

Knowledge Formalization & Representation
Probabilistic Logic, Knowledge Graphs

Conformity Check
Monitoring

4.13.1 Introduction

We study the problem of behavior generation and motion planning. The task of motion planning consists of finding a sequence of configurations of the ego vehicle over time to be passed to a low-level trajectory following controller. In addition to ensuring the safety and comfort of its passengers and of other traffic participants, the planned trajectory should respect the rules of the road.

Current autonomous driving stacks typically structure their algorithms in a pipelined, modular approach, using hand-engineered techniques at the various stages of perception, prediction and planning [84]. While these systems often come with provable properties and strong interpretability, they tend to be brittle and difficult to tune by hand due to information loss between the various components of the pipeline [85]. Recent progress in the field of deep learning promises to ameliorate these issues.

Safety and strong generalization are essential requirements for the homologation and acceptance of self-driving vehicles. In particular, autonomous systems are expected to follow traffic rules and adhere to common safety standards of road traffic at a level that is at least comparable to humans while covering a wide operational domain with a very long distributional tail [86].

We therefore propose to address the question of traffic rule integration into a deep motion planning system. The system is expected to provably follow traffic rules and to strongly generalize to unseen situations.

4.13.2 Use Cases

The system's safety and generalization capabilities shall be demonstrated in simulation, including two common scenes that constitute frequent sources of human accidents

Highway driving. The vehicle evolves at high speeds in a highly structured environment with high interactivity between agents. Cooperatively navigating these environments demands a solid understanding of physical laws and social norms.

Urban driving. The environment is less structured, but highly interactive and governed by complex rules. This is likely to lead to situations of rule conflicts. The system shall be able to detect these conflicts, take sensible actions to resolve them, guided by risk and common social acceptance, and communicate them to the passengers if necessary.

4.13.3 Knowledge sources

The system's knowledge consists of traffic rules and social norms. Legal knowledge might be manually drawn from legal texts like traffic law or learned from expert driving demonstrations. For social norms, learning from human real-world driving data is an obvious choice.

4.13.4 Representation/Formalization concept

Early works have explored the use of logic for reasoning about traffic scenes. Temporal logics have been proposed as a means of reasoning about time series data, which lends itself well to trajectories [87], [88], [89].

To ground the logical symbols, structured scene descriptions are required. Traffic scene ontologies have been developed to solve this task. These can be used to represent static information like the geometry of road segments [90], as well as dynamic interactions between agents, yielding a road scene graph [91].

The motion planning problem under traffic rules can be formulated in a Bayesian framework as the task of inferring a posterior over trajectories given goals and constraints.

One option is to learn the model in a Bayesian fashion by putting a suitable prior over the model and inferring the maximum likelihood posterior model. This typically leads to a regularized optimization problem. Examples for this approach are [92], [93].

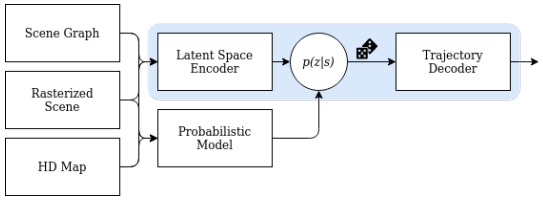


Fig. 39: Sketch of the proposed motion planning system.

Another option is to incorporate probabilistic reasoning capabilities into the model itself. Probabilistic models for rule integration in sampling-based motion planners have been studied in [94], [95]. Building on this approach, we frame motion planning as an amortized variational inference problem.

The aforementioned methods sample a fixed number of discrete trajectory candidates for evaluation using an ad-hoc, hand-crafted algorithm, imposing a restriction on the resolution of the planned trajectories and possibly making the algorithm unsuitable for planning in tight situations like narrow streets or dense urban traffic. To overcome this limitation while keeping the benefits of probabilistic rule integration, we propose to do variational inference in a continuous latent space. Samples can be drawn from this continuous distribution and decoded to trajectory proposals in an end-to-end differentiable manner, with no a-priori restrictions on the trajectory resolution. Inference costs can be amortized using techniques from amortized variational inference, leading to an architecture similar to [96].

A sketch of the model is depicted in the Fig. 39.

4.13.4.1 Conformity check: The probabilistic model is capable of reasoning about the validity of trajectory candidates, yielding an assessment of their legal properties as a side-product. Additionally, the validity of trajectories can be checked in a post-hoc fashion.

While it is often not possible to guarantee complete collision freedom, it can be guaranteed that the ego vehicle is not responsible for collisions if they occur. The necessary rules have been formalized under the RSS framework [97]. An alternative formulation is given by the NVIDIA driving force field [98].

An interesting alternative is given by temporal logics, which make it possible to express rules over time series data. This formalism naturally lends itself well to trajectories. Temporal logics can be converted to automata for online conformity checks or as part of a tree search [99].

Given a structured representation of the environment, a graph of constraints can be set up and pruned [100]. Based on an HD map of the surroundings, it is further possible to set up geometric costs [93], [101], [102].

Finally, the validity of trajectories could be checked by a queryable expert in an interactive fashion using interactive imitation learning methods like DAgger [103]. Recent work has also investigated checking the alignment of planner's implicit value function against a human trajectory preferences for critical trajectories [104].

4.14 FZI Forschungszentrum Informatik (TKS)

Topic overview:

Use Cases (Scenario)
UC 2, UC 3

Knowledge Source
Object properties including velocity, holomonic/non-holomonic properties (Plausibility checks)

Knowledge Formalization & Representation
Possibly Signal Temporal Logic and Linear Temporal Logic schemes, Shape Priors and shape manifold representation via TSDF

Conformity Check
Inference Checkers using online watchers/observer modules, Run Time conformance checkers using STL rules, Object Plausibility using energy based models, Energy priors using product of Experts by combining height and rotation priors

4.14.1 Introduction

There are 2 parts to our study, 1) Motion model with constraints - we try to understand the implicit behavior of non-holonomic vehicles constraints via motion prediction 2) Plausibility checks - we try to create a run-time conformance model on the existing motion model (obtained from first part) to validate the safety assessment without making any design changes to the model.

Motion prediction is the task of predicting the future vehicle states by observing the past state of the vehicles. For such a model it's typical to predict not only a single mode consisting of future waypoints (as an (x,y) state coordinate on a 2D image plane) but rather as a set of modes each with its own probability. In this way one could represent the multi-modality of the environment. The models which are trained this way usually do not encode any physical knowledge about the environments or information about the social norms/cues which are necessary to make informed knowledge. With this we address the problem of using such information (Eg.One cannot/should not drive on the lanes) during training via loss function.

Aleatoric uncertainty about the model can be quantified to certain extents [105] but computation time required for such methods restrict the real time deployment of those modules. On the other hand, Runtime Conformances or Plausibility checks provide explicit solutions through monitoring methods. In case of unbound unsafe trajectory being predicted by Autonomous Systems; monitors can help to switch from unsafe situation by injecting safe actions directly. Such modules provide a safe output trajectory at any given time by having a backup state action which complies to ensure the overall system safety. Checker-doer modules are needed to make the ML based modules deployable for real world applications without compromising on the safety standards.

4.14.2 Use Cases

- Object plausibility detector is used to reduce the influence of False Positives coming out from a 3D object detector. This helps to increase the safety and comfortability of the driving experience. Mostly the data are to be evaluated in urban scenarios where the traffic density is high.
- Runtime conformance module for the trajectory estimator could be demonstrated for Highway pilots exhibiting the merge or cut-in behavior.

4.14.3 Knowledge sources

Motion model requires the environmental information which could be gathered from HD maps where a detailed layered representation is available. Along with this information one would require the RGB information along with LIDAR Pointclouds to make the perception stack provide object detection outputs as can be seen from the works of Pointpillars [106].

Conformance module requires social signals which are not available in traditional manner. For such a module informations such as safety corridors (region around a dynamic vehicle which is maneuverable without causing safety violation) are crucial to make informed decisions. The Module also requires safety rules as could be defined via Linear Temporal Logic (LTL)or Signal Temporal Logic (STL). Example of such definitions can be seen from Nvidia SFF [107], Rulebooks [108].

For object plausibility shape priors in the form of manifolds are necessary. This is required for

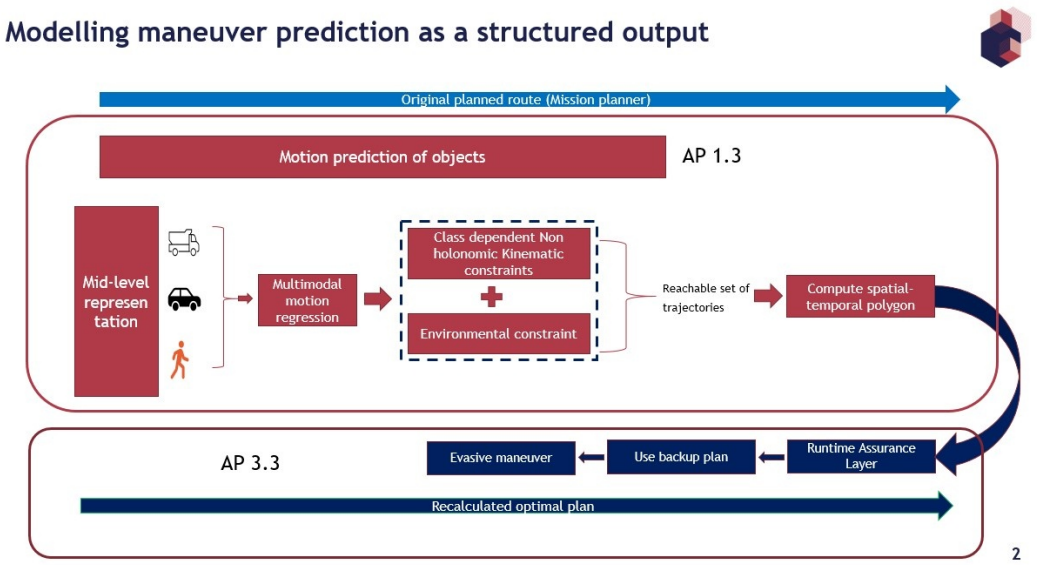


Fig. 40: Sketch of the proposed motion prediction system

computing the energy of the compatibility between mean manifold and detected 3D objects.

4.14.4 Representation/Formalization concept

Motion Model Stack

Plausibility checker for 3D object detector Representing 3D geometry of an object is a very active topic of research within the computer vision and machine learning research community. As the authors of [109] point out, one of the most commonly found 3D object representations due to their ability to approximate arbitrary shapes while offering efficient processing with current graphics hardware are polygon meshes. Datasets such as ShapeNet [110] aim to collect and organize polygonal models and make them available for research.

More attention has been given to volumetric grid representations as they offer the regularity often desired in data processing. One approach to volumetric shape representation is to transform polygons into voxels, as proposed by the authors of [111]. Another way to represent shapes on a volumetric grid is through a Signed Distance Function (SDF) as pointed out by the authors of [112]. The shape's surface is represented implicitly through a continuous volumetric field assigning each point in the field the distance to the closest surface boundary. The sign indicates whether the region is inside (-) or outside (+) the object. In [112]the authors use a neural network to learn a generative model that can produce a continuous SDF. While continuous SDF are not bound to a volumetric grid representation, a variation called Truncated Signed Distance Function (TSDF) as Engelmann et al. use in their work [113] has been proposed.

TSDF allows shapes of object instances to be homogeneously approximated in a voxelgrid representation and can be seen as a discretized version of a continuous SDF. This is achieved by discretizing the volume around an object shape instance into a regular axis aligned voxel-grid and assigning the corresponding SDF values to each voxel's vertices. Such proposed representation for the objects could look as shown in the image below.

4.14.5 Conformity check

Plausibility checker for 3D object detector Several works exist which aim to verify the detected objects

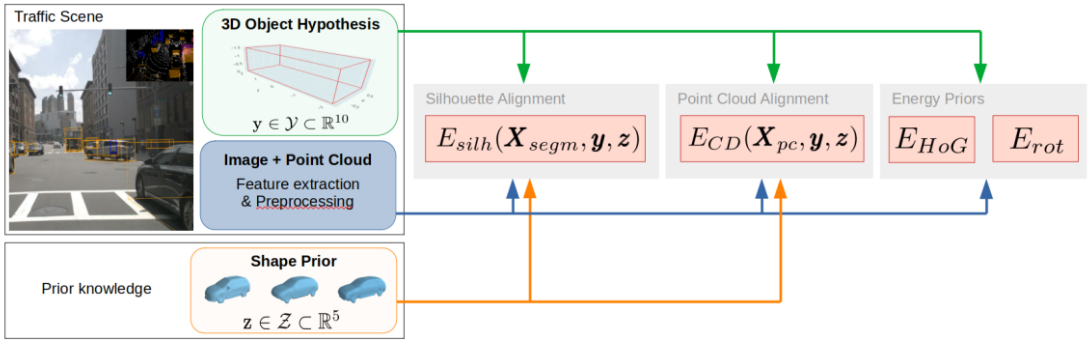


Fig. 41: Proposed concept for Energy based Plausibility checker

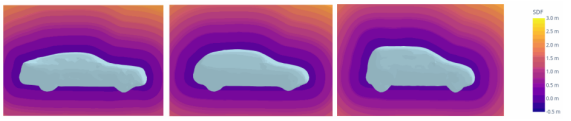


Fig. 42: The colored plane shows the respective points described through Signed Distance Function

existence in a fusion system. Often the Dempster-Shafter theory of evidence (DST)[114] is used to implement and combine plausibility features on various system levels. A powerful data-driven way of including prior knowledge in a detection system is to have data on an object's physical spatial appearance; its 3D shape. However several ways exist in which the data of a 3D shape can be represented. Approaches are needed that allows one to parameterize and easily handle variations, which could arise from perspective differences and measurement noise of sensors. Energy based models are an essential part of this work. It uses energy functions to measure the compatibility of observed data and assumptions from prior knowledge with object detection hypotheses. The usage of four different energy functions and priors is proposed (as can be seen from the image below).

Each individually encodes a different data-prior knowledge requirement. For example, HeightOver-Ground (HoG) energy prior in the image provides us a high energy value if the vehicle is above certain height from the estimated ground plane. By combining these four energy functions we lay the concept for generating an un-bounded energy value. As LeCun et al. point out this requires special care as opposed to probabilistic models, the energy functions are un-calibrated. The native approach to combining multiple energy functions is through a linear combination. Goodfellow et al. show that this native combination can be related to the Product of Experts approach proposed by Hinton[115].

4.15 OFFIS e.V.

Topic overview:

Use Cases (Scenario)
UC 1, UC 2

Knowledge Source
Knowledge on Scene, Knowledge on Evolution

Knowledge Formalization & Representation
Traffic Sequence Charts (TSCs)

Conformity Check
Timed Automata

4.15.1 Introduction

In this section, the concepts for knowledge representation and conformity check are described that are based on OFFIS' own *Traffic Sequence Charts* (TSCs) [116], a formal description language to describe abstract scenarios. This language is able to capture information on traffic scenarios, like the number of lanes, traffic participants, and evolution over time with a formal interpretation and sequential in the background. According to Figure 43 OFFIS uses the TSCs to store two types of information: On the one hand scenario and use case information where knowledge is available, on the other hand, the available knowledge itself. This extends TSCs to knowledge building blocks to extract the knowledge later on in order to support AI training and to extract monitors for knowledge-conform AI observation.

In the following subsections, we describe the use cases and relevant knowledge we want to integrate into TSCs in 4.15.2 and 4.15.3 first. Then we describe how to formalize and represent the knowledge in TSCs in 4.15.4 concluding with a concept for conformity check in 4.15.5.

4.15.2 Use Cases

As part of the KI-Wissen project, OFFIS e.V. is focusing on the overtaking maneuver. Here, it was decided that an increasing complexity of the use cases should take place within the project period. This leads to the fact that all planned concepts can first be developed fundamentally as a proof of concept and consequently be extended in parallel and iteratively in the manageable complexity of the use cases.

In the first use case developed, a car following maneuver is considered in the context of an adaptive cruise control (ACC) system. This use case corresponds to the first and lowest complexity. Mainly longitudinal dynamics and distances to the vehicle in front are relevant. The goal is to maintain a safe and optimal distance from the vehicle in front. A static representation of the use case is presented in Figure 44 where the blue vehicle equipped with a prior knowledge trained AI component follows the black front vehicle.

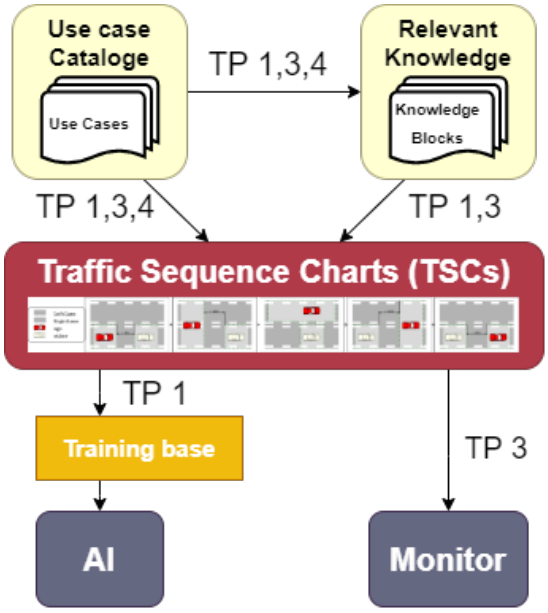


Fig. 43: OFFIS' approach using TSC as knowledge building block capturing knowledge and use case as AI training and monitor base.

In the second use case, a first overtaking maneuver is considered based on the ACC use case. This use case corresponds to the medium complexity. In addition to longitudinal dynamics and distances, lateral dynamics and distances are also included. Furthermore, an oncoming vehicle will additionally increase the complexity. The goal, is to maintain a safe and optimal distance from the vehicle in front. A static representation of the use case is presented in Figure 45 where the blue vehicle equipped with a prior knowledge trained AI component has the goal to overtake the black vehicle in front. The red vehicle corresponds to the oncoming vehicle.

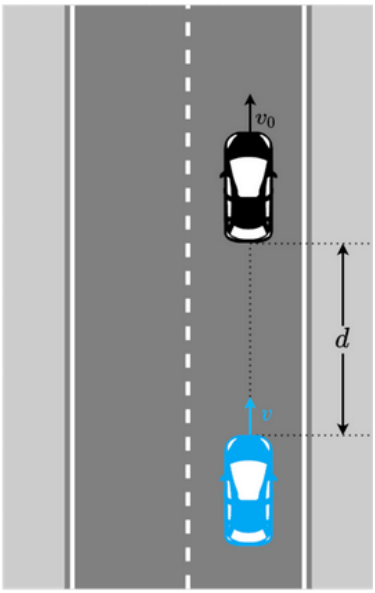


Fig. 44: OFFIS' developed Adaptive Cruise Control (ACC) Use Case

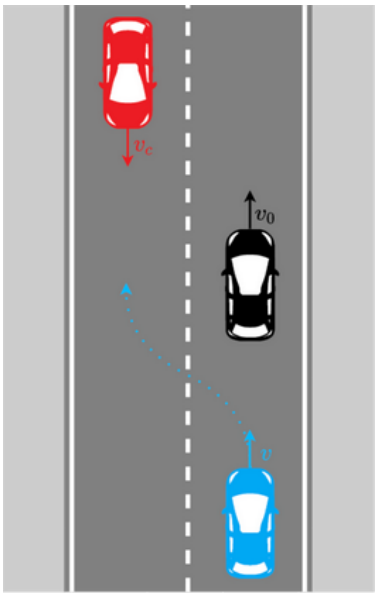


Fig. 45: OFFIS' developed overtaking Use Case

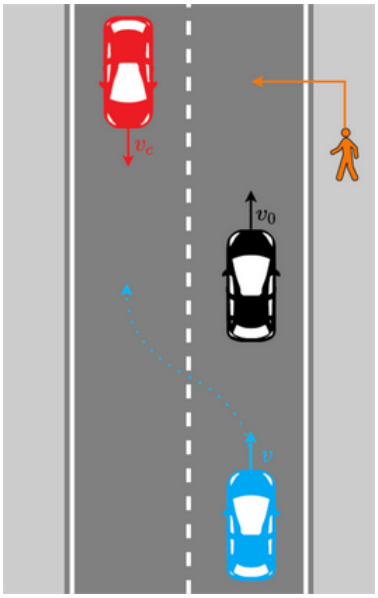


Fig. 46: OFFIS' developed occlusion Use Case

In the third use case, based on the previous overtaking maneuver, the complexity is increased by a pedestrian on the sidewalk, which potentially crossing the street or is occluded by the black vehicle. This use case corresponds to the highest complexity. Additional dynamics for pedestrians are included, which introduce many uncertainties and thus higher complexity for a safe overtaking maneuver. The goal, as in the use case before, is to perform a safe, efficient, and pleasant overtaking maneuver, while additional traffic participants and their different dynamics are taken into account. A static representation of the use case is presented Figure 46 where the blue vehicle equipped with a prior knowledge trained AI component has the goal to overtake the black vehicle in front. The red vehicle corresponds to the oncoming vehicle and the orange road user corresponds to the pedestrian. Different variants can be derived in the context of this use case. For example, the front vehicle can be a parked bus, a parked vehicle, or a moving vehicle.

The developed Use Cases are based on the use cases 1.2, 1.3, 2.3, 2.4, 3.2, 3.6 and 3.14 already created in the project.

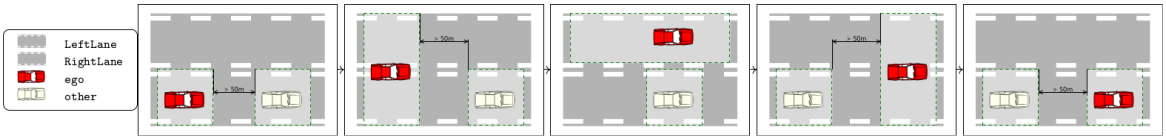


Fig. 47: TSC describing an overtaking maneuver.

4.15.3 Knowledge sources

Now that Use Cases are described in 4.15.2, the relevant knowledge still needs to be identified according to Figure 43.

As there is a lot of knowledge to know by various stakeholders for every Use case, not everything is intended to be formalized and used through TSCs later on. To decide, which of that knowledge is even relevant, we identified observable and influenceable variables of an agent vehicle, first. In the ACC-UC (44) observable variables are the velocities of the vehicles as well as the distance between them. The influenceable variables in this use case is the acceleration of the ego vehicle.

For observables we derived two types of relevant knowledge: First, there is the knowledge that constrains the values of variables. This can be physical measurements like the size of a vehicle, legal norms like the maximum allowed velocity, or suggestions from experts like the maximum acceleration to ensure a human-conform drive. Second, there is knowledge, that closes the gap between the variables and the goal of the use cases. As the goal in the ACC-UC is to keep a safe distance to the vehicle ahead, the relevant knowledge describes the physical dependency between acceleration, velocity, vehicle distance, and a safe time headway.

In Figure 48 the derived relevant knowledge for the occlusion Use Case is shown. We classified the knowledge into "knowledge on scene" that describe limitations and valuations of scene describing variables and "knowledge on evolution" that describes knowledge on how variables will be influenced and evolve over time.

4.15.4 Representation/Formalization

As the use cases are described and the relevant knowledge is identified in the previous sections, now both should be integrated and formalized via the *Traffic Sequence Charts* (TSC) language. First, we will describe the TSC language to explain the anchor points for knowledge formalization later on.

In Figure 47 an overtaking scenario is described via a sequence of images, called invariant nodes. Each invariant node describes constraints for a traffic situation that hold for a while. For example, the first invariant node specifies, that there is a red car behind a white car on a two-lane street, and the possible spaces, where each car can be located, are at least $50\,\mathrm{m}$ away from each other. Note that an invariant node only describes constraints for a traffic scene and not a concrete instantiation. In the first invariant each car can be close to the middle lane or far away, both cars can be $60\,\mathrm{m}$ or $800\,\mathrm{km}$ away from each other and there can be other cars on the left lane.

Invariant nodes can be sequenced after each other to describe a flow of traffic situations to a scenario. The TSC in Figure 47 describes that there are five phases in an overtaking scenario. First, there is a phase where a red and a white car drives together on a right lane with a minimum distance in between, then there is a phase where the red car can additionally drive on the left lane, then the red car is completely on the left lane with a larger horizontal free space than the white car, than the red car is ahead of the white car and can drive on both lane again but with a minimum distance of $50\,\mathrm{m}$ again, and finally, both cars are again on the right lane, but now the red car is at the front. Note that neighbored invariant nodes must not exclude themselves.

To enable a uniform interpretation of those symbols, which occur in graphical invariant nodes, the meaning of each symbol is introduced in a *Bulletin Board* on the left side. Each symbol represents an element from the TSC's underlying world model, that defines the object involved in the scenario. The objects are equipped with properties like width and driving direction for road objects or size, velocity,

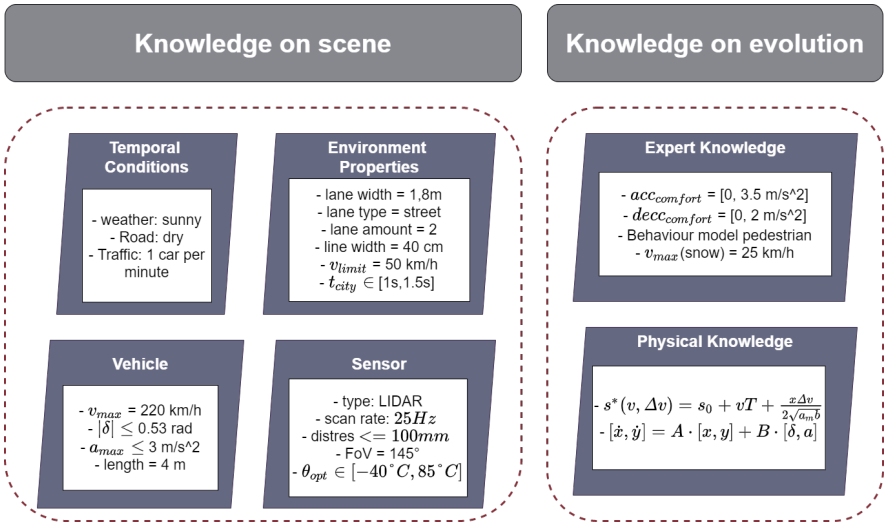


Fig. 48: Relevant knowledge for occlusion Use Case (46 classified in first stages of knowledge blocks.

and acceleration for car objects. As these properties can be restricted via constraints, knowledge on scene and environment properties like the maximum allowed velocity on a driving lane or the steering range of a car can be anchored in the TSC's world model. Also, more complex constraints including multiple properties are possible. Here there is an anchor point for mathematical/physical knowledge, as properties like driven distances, velocity, acceleration, and orientation, etc, can be constrained according to a vehicle motion model. All of these constraints in the world model are valid worldwide for every invariant node.

In Figure 49 you can see a hexagon after the bulletin board, which is called premise and acts as the necessary condition for the consequence of invariants. The left part of the premise is called history and specifies, what has to be valid in the past for the necessary condition. The right part of the premise is called future and specifies, what has been observed in the future for the necessary condition. Together with the consequence, history, and future reveal the following type of propositions: If the history has been observed in the past and the future will be observed, then the consequence must hold.

Hence, the TSCs can specify safety conditions as explained with the following example in Figure 49:

If *(History conditions)*
the red ego car drove with a higher velocity than another white car and the distance between the cars was greater than a $d_accFollowingActive$ and smaller than a d_view distance for a while

and *(Future conditions)*
there will be no further car between the red and the other car drives with limited velocity and acceleration in the future

then *(Consequence conditions)*
then the situation has to behave as follows: First the situation is like in the history, then the distance between the two cars will be smaller than $d_accFollowingActive$ and greater than a target distance for a while, and finally the distance is close to a target distance (with regard to a tolerance epsilon) but still greater than a safety distance.

Finally, we can use the TSC-language to formalize knowledge on object's properties via world wide valid constraints (world model) as well as a sequence of constraints (consequence consisting as invariant nodes). For mathematical and physical knowledge and laws, the world wide constraints are the canonical candidates, as this kind of knowledge

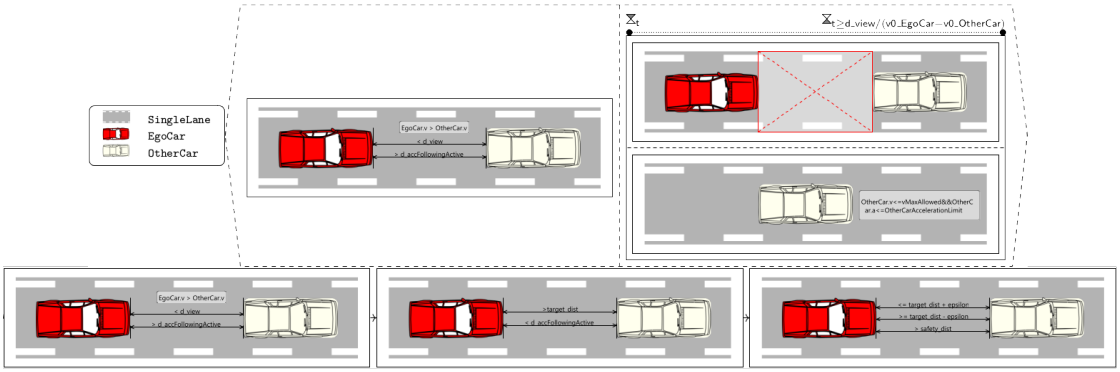


Fig. 49: TSC describing an ACC Scenario for Lane keeping and distance holding.

belongs to hard facts of the world.

4.15.5 Conformity check

Based on the presented specification *Traffic Sequence Charts (TSC)*, a so-called monitor should be generated, which observes the conformity of the artificial intelligence concerning the integrated prior knowledge. The final goal is the automatic generation of a monitor from the TSC specification with subsequent code synthesis and instrumentation in a simulation environment for runtime monitoring.

Using the work from the subproject *Knowledge Integration* regarding knowledge formalization and knowledge representation, the following concept assumes that the world model and ontologies by the TSCs as well as the integrated prior knowledge into the artificial intelligence are correct.

In order to achieve the aforementioned goal, research was first conducted to find a suitable monitoring method. Both declarative and directly executable approaches for a monitor were considered, taking into account the implementable set of specifications by TSCs. Declarative approaches in the context of our investigation are formal languages. One property of formal languages is the possibility to establish different levels of abstraction. However, the developed monitors must be synthesized into executable code [117]. In the context of monitoring, formal languages often form an abstraction of the actual system behavior and convert specifications into so-called events. A specification often describes how a sequence of events (traces) may be executed to fulfill the specification [117]. To verify conformance, the so-called trace of the system is observed and it is checked whether the trace complies with the specification. If a specification is not met, there is no complete conformance to the integrated prior knowledge. The possible formal languages differ concerning the possible expressiveness. In particular, a distinction is made between continuous and discrete systems as well as in the way temporal aspects are expressed with respect to finite and infinite system executions [117]. Among formal languages, Linear Temporal Logic (LTL, and derivatives) [118, 119], Signal Temporal Logic (STL) [120], and Interval Temporal logic (ITL, and derivatives) [121] were considered. Due to the possible temporal specifications by a TSC and the respective properties of the temporal logics, $TLTL_3$ [119] derived by LTL is the most promising to achieve the stated goal.

In addition to the declarative approaches, directly executable approaches were also considered. Here, only finite-state machines (FSM) were referred to. FSMs operate with error states if an executed action of the system does not match the defined behavior of the monitor. The advantage of FSMs is their direct executability. Unlike formal languages, however, different levels of abstraction can only be reached with effort [117]. Similar to the examination of the formal language, the set of possible specifica-

tions by TSCs was also considered when analyzing FSMs. Due to the temporal specifications by a TSC, the focus of the investigation was quickly on timed automata. Concerning possible specifications to hard constraints (speed limit, safety distance) but also soft constraints (e.g., "urban road: a car should not drive to slow"), which often arise from fuzzy rules, a Probabilistic Timed Automata [122] seems most promising to achieve the set goal.

Apart from the choice of the monitoring method, the extraction and synthesis of the specifications in the TSC to a monitor must also be investigated in the course of the project. Furthermore, the instrumentation of the monitor in a simulation environment has to be answered [117]. In the sense of the first proof of concept, based on the mentioned low complexity use cases, a timed automaton is first created as a monitor and integrated into the intended ecosystem.

For the extraction of specifications from the TSC, the first step is to consider hard constraints and soft constraints. Furthermore, important properties related to the use case under consideration are to be included from the world model and the ontology. Depending on the goal of the use case and the resulting specifications, one monitor can be generated for one specification or one monitor for several specifications. In addition, various metrics can be included with thresholds. Due to the TSC semantics, it is possible to build one state per invariant node. As soon as a new state is reached, the specified constraints apply, which are defined by the specifications within the TSC.

The instrumentation of the monitor into the CARLA simulation environment is planned. There are first approaches for code synthesis of temporal logics into the environment of CARLA, which allows the consideration of different abstraction levels. The approach from [123] can be seen as a starting point for further developments. The code synthesis of a timed automaton is in principle simpler [117, 124, 125]. The instrumentation of synthesized automata in CARLA does not initially represent a major effort. However, if the goal of runtime monitoring is considered, threading patterns must be considered for multiple automata.

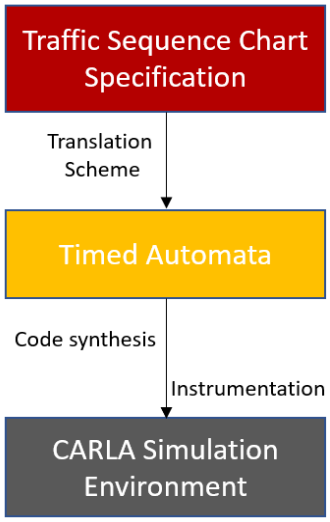


Fig. 50: OFFIS monitoring concept using TSC specification to generate timed automata followed by the instrumentation within the CARLA simulation environment

Here, a distinction can be made between asynchronous and synchronous execution of the monitor for simulation. An asynchronous execution can mean a decisive performance advantage, whereas synchronous execution can benefit from real-time observations [117]. In the context of the project, based on the assumed correct ontology and specification on which the simulation build upon, the asynchronous execution is chosen.

In conclusion, as presented in 50, the specifications within the Traffic Sequence Charts will be the foundation for the timed automata. Firstly, a translation scheme has to be developed to be able to transform TSC specifications into timed automata. Afterward, the code synthesis of the timed automata to executable source code takes place. Finally, the challenges mentioned before regarding the instrumentation within the simulation environment will be addressed.

4.16 Robert Bosch GmbH

Topic overview:

Use Cases (Scenario)
UC2, UC3
Knowledge Source
Requirements from customer and legal documents (Regulations, Standards, Traffic Rules)
Knowledge Formalization & Representation
Text embeddings, ontology / knowledge graphs
Conformity Check
No involvement

4.16.1 Introduction

This section concentrates on knowledge extraction and knowledge representation relevant within a software and product development process. Tier 1's, such as Bosch, have to deal in general with different customers in different target markets. However, Tier 1's aim to develop solutions which can be reused to a certain intent. Thereby a key aspect is to understand and detect similar, revised or new requirements based on customer wishes and legal regulations. Since Tier 1's (and OEMs) have already implemented customer and legal requirements in existing products in the past, a knowledge base of known and implemented requirements is available for Tier 1's.

Tier 1's do not start from scratch. Rather the extraction, representation and integration of the new or revised knowledge is the essential and crucial part. The main idea for KI Wissen is to extract and to represent this "delta knowledge" with the power of natural language processing methods and knowledge engineering approaches.

Figure 51 shows the general approach to extract knowledge from text and integrated it into software. The following sub sections will explain the approach in more detail.

4.16.2 Use Cases

With regard to the use cases we concentrate on the use cases were different target markets and different customers have the most influence on functional behavior. This is for us use case 2 and use case 3. There we aim to detect, extract and represent customer and target market specific parameter values and conditions from textual input which can be integrated in decision making and trajectory planning.

4.16.3 Knowledge sources

Tier 1's, such as Bosch, have to deliver safe and secure components for a overall system. Therefore guidelines and processes have been established industry wide to foster a high software and product quality. The most common process for software (and product) development is the V-Model approach.

In the V-Model approach requirements are the key element. Requirements mainly describe in a textual way functional behavior and test cases. Figures and tables additionally support the textual description. Thus knowledge for a Tier 1 is represented by various sources. They consist of customer requirements, internal requirements and requirements from legislators.

Requirements are typically managed in a requirement management system (e.g. IBM Rational Doors) for further processing and to ensure traceability. They are documented in tables, i.e. structured data and enriched with meta data, so called attributes and mostly follow a (simple) schema.

Lately there has been some research by [126] to extend this schema to a semi-structured language. This can be seen as a intermediate step towards a formalization of requirements and would simplify the next steps.

4.16.4 Representation/Formalization concept

The extraction of delta knowledge and its formalization means first of all the comparison with known requirements, i.e. well known and already implemented knowledge. For this purpose, a concept was developed that condenses the extraction and representation of delta knowledge over several steps. These four steps are:

1) Identification of context (high level)

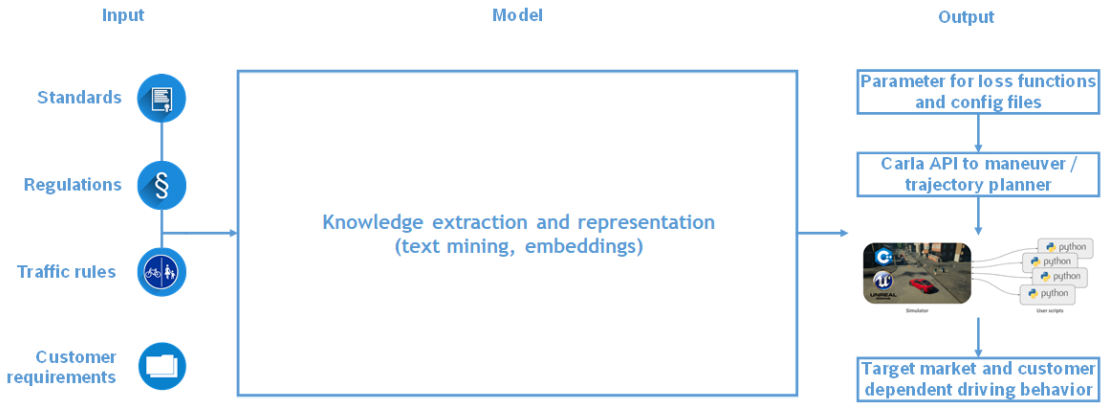


Fig. 51: General approach (high level)

2) Identification of delta knowledge and knowledge linking
3) Semantic analysis and knowledge representation
4) Integration of knowledge

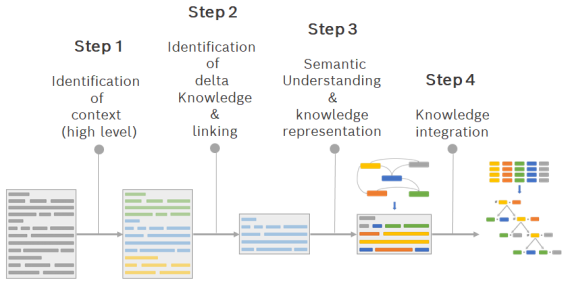


Fig. 52: Step-wise extraction and representation of knowledge out of requirements

1.Identification of context (high level): The first step is a filtering and categorization step. Requirements typically cover definitions, functional behavior and test cases with regard to the overall system and to specific function blocks of the overall system. Thereby the overall System is basically covered by OEMs and function blocks from Tier 1 and OEMs. Function blocks are for instance lane keeping, lane change, emergency braking, ACC or video perception. Consequently for a Tier 1 it is sufficient in next steps (formalization / knowledge representation) to only consider a subgroup of relevant function blocks as part of their product portfolio. Hence the first context information required is associated to the following:

- Which function block is involved?
- What is the content?
- Who is responsible to full fill the requirement?

From a meta data / attribute perspective we have three classes:

- Function ID: An Integer, correlated to a function block
- Requirement type: A String, describes what type of requirement it is: Test, Function or Definition.
- Relevance: A String, describes for whom the requirement is relevant, OEM or Supplier

For the enrichment of the requirements with these three attributes we handle the problem as a classification problem. For this purpose a classification model has to be trained on historical data as illustrated in figure 53.

Historical data are accepted requirements from previous customer projects and already analyzed legal documents / requirements. Since different customer and different legal authorities use different words and terms the training highly depends on the input text document. Therefore it makes sense to train input specific models and store these models in a model store (i.e. fileshare of trained AI models).

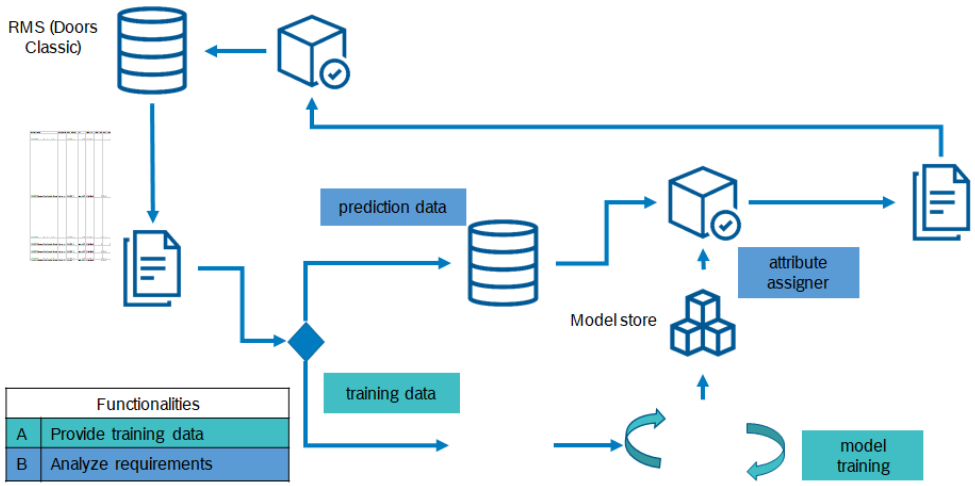


Fig. 53: Overview of the metadata enrichment approach

Figure 54 explains the approach in more detail and highlights the applied software packages.

First of all we implemented a light weighted ETL (Extract-Load-Transform) process. The ETL process extracts tables from DOORS as csv files, which can be read by the pandas package to make the tables as dataframes accessible by other AI / NLP python packages. After this data preparation, we start with tokenization of our requirement object text. Tokenization refers to the decomposition of texts into indexed words/tokens. It is a mandatory step for the extraction of further features which are forwarded to the classification models. Such features can be embeddings (e.g. n-dim vectors) or word counts. Figure 55 shows the architecture of our nlp pipeline in more depth.

The concept of tokenization involves:

1) Pre-tokenization of text before actual tokenization
   - e.g. remove spaces, convert umlauts, use vocabulary
2) Build a domain-specific dictionary
   - Integration of typical signal names/units, formula symbols
   - Integration of typical spelling errors
   - Integration of typical synonyms
3) Post-tokenization after actual tokenization
   - merge tokens, e.g. brackets and word

After the tokenization the tokens are used to build a vocabulary to target domain specific language aspect. With the vocabulary we are able to build a customized language model. This concept is based on the OSS software package spacy (https://spacy.io/) and the word2vec method.

For the classification model itself we do not aim to develop new classification methods rather to benchmark different state-of-the-art. The following methods were applied and implemented: a Linear Model, a Bi-LSTM Model and a CNN. The application of different methods is motivated by two facts. Firstly, different attributes might build cluster with different characteristics and sizes. Secondly customer and legal authorities have their own specific words and terms. The sensitivity of the classifiers to this issue might be different.

As shown in figure 54 the handling of the different trained models requires a smart mechanism to store and to deploy the different models. This leads to the need of a platform / infrastructure to manage different trained models / classifier. Commonly referred to as Machine Learning Life Cycle Management. For this purpose we use the

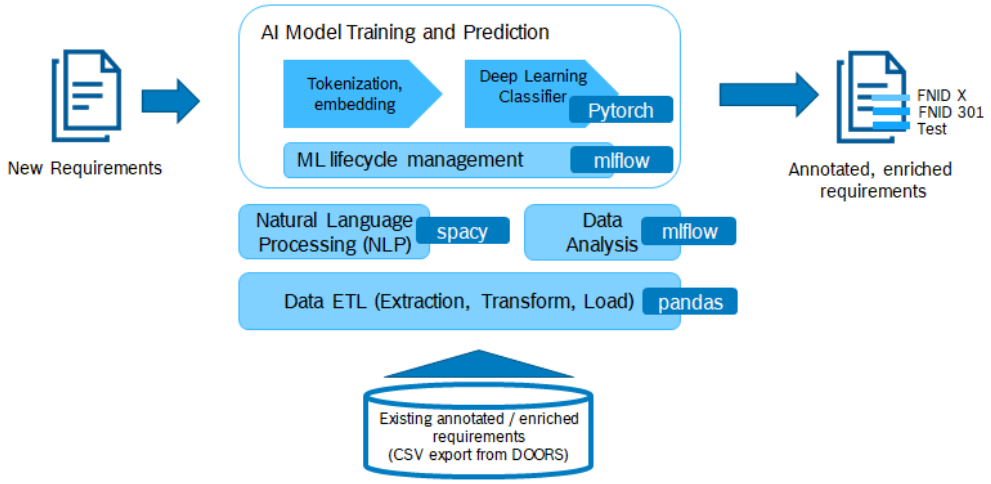


Fig. 54: Overview of the applied technologies for the metadata enrichment

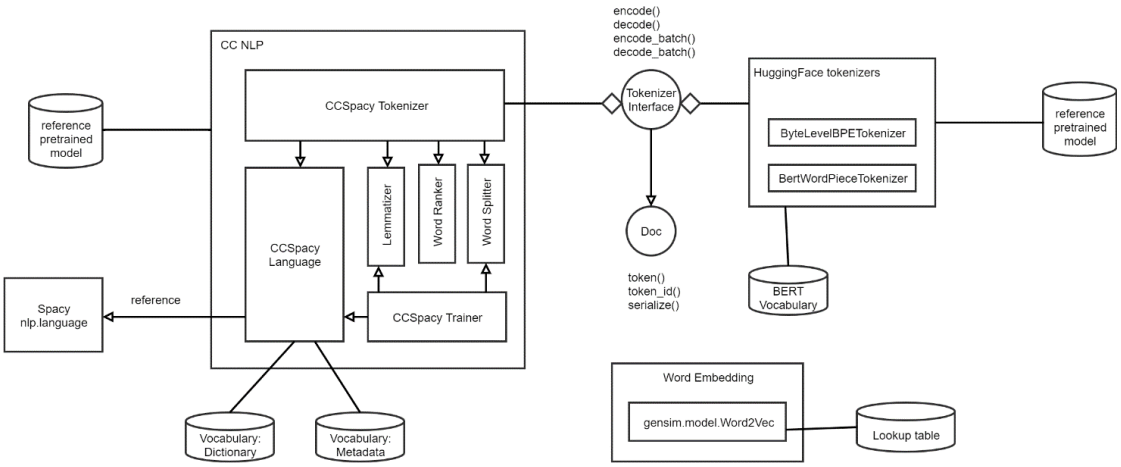


Fig. 55: Overview of the NLP pipeline

OSS package mlflow. Mlflow allows to iteratively optimize model parameters and to deploy models.

As an outlook for the feasibility of our concept we present first results. Figure 56 shows first results for the function ID attribute tested on customer requirements. Without a deeper analysis we see that different methods perform quite similar, independent from the complexity of the method. Further research on the text corpus is planned to better understand the result and to further improve the accuracy. We will also investigate mechanism to fine tune language models to different domains (legal vs. customer). Language models such LegalBert could be beneficial in the future.

2. Identification of delta knowledge and knowledge linking: The second step checks within the pre-filtered requirements from step one if requirements contain new or revised knowledge for certain function blocks. For this reason we apply algorithms which are capable to detect near duplicates, i.e.

Model	Macro F1 / Accuracy
CNN + ccspacy tokenizer	0.5511 / 0.8027
TFIDF + Linear + ccspacy tokenizer	0.5457 / 0.8225
Embedding + Linear + ccspacy tokenizer	**0.5654** / 0.8112
RegBi-LSTM + ccspacy tokenizer	0.5638 / **0.8297**

Fig. 56: First analysis result for attribute type function ID

requirements which share similar content. For that reason we benchmark different approaches and metrics. We distinguish between lexical similarity and semantic similarity. For lexical similarity, i.e. sentences share a large amount of same words / tokens, we use a well known method / metric from auto correction - The Levenshtein distance.

However this methods is not well suited for the detection of semantic similarities, i.e. if two requirements describes the same content with different words or in a different language. Therefore we also consider the language models from step 1 to transform sentences into embeddings. Once the embeddings are built we apply locality-sensitive hashing (LSH) methods such as minHash to calculate the distance/relatedness to vectors. LSH approaches are well suited for comparison a large amount of requirements, since they are computational efficient. Especially for the large number of customer requirements this is an important aspect. For some text documents types (regulations) we assume that lexical similarity could already be a sufficient. For others such as traffic rules / requirements we look more into the direction of semantic similarity.

3. Semantic understanding and knowledge representation: After step two we have identified new or revised requirements for specific function blocks. However we have not yet gained a deeper semantic understanding of the content. This is done in step three. Step three targets the semantic understanding and knowledge representation of the requirement. This is a quite sophisticated step. It requires taxonomies / thesaurus / ontology's to describe the concepts and relations and data annotation techniques to find relevant word / terms from the taxonomy or ontology within the requirement.

As the state-of-the-art analysis in AP5.1 pointed out there is not an ontology which fits directly to customer an legal requirements. We plan to start with a light weighted taxonomy and ontology for a specific function block (= application ontology) and work towards a domain ontology. In general we want to extract signal names, signal values, signal ranges with regard to environmental and traffic conditions, field of applications and dependencies.

For the data annotation part two approaches are possible. We can use an annotation tool such as inception to build training data for text classifier and / or we can use heuristics together with packages such as snorkel to classify texts based on rules. The extracted knowledge can be represented in knowledge graph or depending on the complexity in linked tables.

4. Integration of knowledge: For the integration of knowledge we begin with a more simple approach. We use the knowledge representation from step three to generate a parameter database. This parameter database will depend on the target market, the customer, the traffic condition and environmental condition etc. The parameters can be integrated in a loss function for the trajectory planning, configuration files for trajectory and decision planning and to design an additional decision tree in a late fusion step. More sophisticated approaches such as semantic reasoning and knowledge graph embedding will be investigated as well.

4.17 Universität des Saarlandes (UdS)

Topic overview:

Use Cases (Scenario)
UC 2 & 3
Knowledge Source
Traffic law
Knowledge Formalization & Representation
Logics & special mark-up languages
Conformity Check
Evaluating the legal conformity of autonomous maneuver decisions (ex ante and ex post)

4.17.1 Introduction

In TP 1 and TP 3 the University of Saarland (UdS) will focus on the integration of normative knowledge into the AI. The group will also be working on approaches to give the respective systems the ability to check its own maneuvers for compliance with given road traffic regulations (conformity check). Part of the conformity check will also be the ability to handle conflict of norms and act according to rule exceptions. A major implementation-related challenge of integrating normative knowledge into AI-Systems will be the formalization of the said knowledge. One of the key aspects why this is challenging is because a lot of rules are written as general clauses or use indeterminate legal terms which need a judicial interpretation in the specific case. E. g. see § 1 I StVO: it is ruled that "Participation in road traffic requires constant caution and mutual consideration". Another example would be § 3 I StVO: "A person operating a vehicle may only travel at a speed that allows them to be in constant control of their vehicle. In particular, they must adjust their speed to road, traffic, visibility and weather conditions as well as to their personal abilities and to the nature of their vehicle and its load". The question in real life situations regarding these norms would be "what is the correct adjustment for the specific situation?" or "what does mutual consideration mean in specific real-life situations?". A human operator would usually learn basic principles and standard situations and transfer the knowledge to handle those situations on other similar circum-stances or use his experience to know under which circumstances the vehicle speed must be adapted (e. g. slowing down when it is raining heavily). A knowledge-based AI does not have this ability. Thus, the overarching goal is to describe all behavioral requirements that can be induced from the relevant norm in an unambiguous, machine-interpretable way. Moveover, the ability to check the maneuvers on rule compliance is vital as well. The legal obligation to put the AV in a safe state autonomously if the continuation of the journey is only possible with the violation of road traffic rules, § 1e II no. 3 StVO. Although road traffic rules being rather straightforward and precise there still may be situations where the given rules are insufficient to handle these situations. An example would be the occurrence of a conflict of norms (one norm rules the opposite of another rule while both rules usually would be applied).

4.17.2 Use Cases

The University of Saarland will focus on those Use Cases which will provide the most in-depth legal issues and challenges. This is why Use Case 1 is not overly interesting for these questions because the overarching rule in this Use Case is to not endanger the pedestrians. Here, the legal rules are quite clear that a collision with a pedestrian is not acceptable. Use Case 2 and Use Case 3 on the other hand raise profound legal questions that are related with the problems of defeasibility and tradeoffs in between legal norms, which we particularly want to address. That's why will focus our work on these two Use Cases.

4.17.3 Knowledge sources

The main sources for our relevant knowledge are written legal road traffic rules. Thereby the most important document for is the StVO because it establishes the most important rules of conduct for the specific road traffic scenarios. But a lot of these rules need clarification and beyond that not all road traffic scenarios can be anticipated in every detail by written legal norms. That's why another important source of knowledge are court rulings and commentaries of road traffic rules in literature. Court rulings specify certain rules for specific situations and help

	Traffic light violation	Speeding	Safety distance violation
Level 1		First degree	
Level 2	First degree	Second degree	First degree
Level 3	Second degree	Third degree	Second degree
Level 4	Third degree		Third degree

Fig. 57: Table displaying the degrees of infringement and corresponding levels

to clarify road traffic rules, for example those rules which are written as general clauses. For unclear norms for which there are no judgments yet it is to use legal commentaries and the general rules for the interpretation of legal norms to gather the relevant knowledge.

4.17.4 Representation/Formalization concept

A concept for the formalization of road traffic rules will include the usage of LegalRuleML. In this context we will provide all the relevant legal knowledge to project partners who work with LegalRuleML. This will include the interpretation of the relevant rules regarding the specific road traffic situations.

4.17.5 Conformity check

For now, we want to focus our work regarding the conformity check on the ability to assess road traffic rules and thereby gaining the ability to handle conflict of norms. Our first concept therefore is a matrix in which certain road traffic rule violations are assessed and categorized.

In this (preliminary) example described by the table in Figure 57 a first-degree violation would be the most severe violation of road traffic rules and must be avoided under any circumstances. The severeness of the violation decreases the higher the number of the degree of the violation is (first degree violation is more severe than a second-degree violation). In another step certain rule violations will be assessed in terms of danger for other road traffic participants and effect on the flow of traffic and categorized in several degrees. For example, the higher the speed of a vehicle regarding a speeding violation, the higher this behavior is categorized in the assessment of the severity of rule violations. This assessment would allow to handle conflict of norms in a formalized way. Furthermore, certain rule violations could be given abstract value regardless of the severeness of the violation in the specific situations. For example safety distance violation would be given the value of 75 while traffic light violation would be given 90, meaning that the compliance with traffic lights is generally more important than keeping the correct safety distance. With this assessment it would be possible to handle conflict of norms if the specific rule violations would be on the same degree. Please note that those assessments in the examples are all of preliminary nature and only created for the purpose of explaining the concept. For instance, one has to consider to move away from a discrete space (3 levels) in which the gravity of an infringement is measured, to a continuous space (e.g. a real number between 0 and 1. Furthermore, it needs to be clarified if such an tiered assessment could be used with LegalRuleML.

- Traffic light violation
  - First degree = passing a red traffic light at least $3\,\mathrm{s}$ after it became red
  - Second degree = passing a red traffic light at least $1\,\mathrm{s}$ after it became red
  - Third degree = passing a yellow traffic light
- Speeding
  - First degree = driving at least $50\,\mathrm{km/h}$ above the speed limit
  - Second degree = driving at least $25\,\mathrm{km/h}$ above the speed limit
  - Third degree = driving at least $10\,\mathrm{km/h}$ above the speed limit
- Safety distance violation
  - First degree = falling below the required safety distance for more than $7\,\mathrm{s}$
  - Second degree = falling below the required safety distance for more than $5\,\mathrm{s}$
  - Third degree = falling below the required safety distance for more than $3\,\mathrm{s}$

4.18 Valeo Schalter und Sensoren GmbH

Topic overview:

Use Cases (Scenario)
UC2
Knowledge Source
Object models, Sensor hardware
Knowledge Formalization & Representation
Mesh grid, point cloud, GMM, table, YAML file
Conformity Check
Comparison with object predictions & earlier classifications

4.18.1 Introduction

Valeo contributes in two research topics to knowledge-based evaluation of LiDAR data:

1) detection of optical misalignment of Scala sensors
2) plausibility checks for three dimensional perception via object model matching

Therefore the following sections are divided accordingly.

4.18.1.1 **Optical misalignment of Scala sensors**:
Autonomous driving is highly depending on environmental sensors and their reliability. They have to detect the traffic, obstacles and vulnerable road users with high confidence in order to enable the vehicle to induce the right maneuvres. Up to now deep learning for LiDAR data is not yet as advanced as e. g. for camera images. However, LiDAR sensors are important not only for autonomous driving, but also in the fields of computer vision and robotics. Algorithms have been developed for classification, object detection, instance/semantic segmentation and tracking, similar to the classical image processing [127].

Due to many years of experience with the *Valeo Scala* sensors, several ageing processes have been detected, which might arise during the long period of operation in a car. One of the problems is the misalignment of the optical parts of the sensors due to, e. g. evaporation of solvent from the glue which is used to fix the lenses, or vibrations during operation in the car. It has been possible to detect this misalignment on a test bench, but not during normal operation of the sensor in a car.

The reliability of the sensor misalignment detection is crucial: False negative detections compromise safety, while false positive detections cause high costs and dissatisfaction of the customer.

4.18.1.2 **Object model matching**:
The task of 3D object detection is a crucial part in the pipeline of autonomous driving. The correct operation of the subsequent modules such as motion prediction, planning and steering depends on the quality of the perception results. Therefore, Valeo aims to evaluate and check plausibility of the perception results. Thus, the reliability of the perception results shall be increased for further modules. To realize such a plausibility check of the results of a 3D object detector object models shall be utilized as a-priori knowledge. By comparing the predicted objects with these given object models a decision whether to discard or keep the prediction is taken. Since the focus is set on car objects, Valeo will apply this work mostly on use case 2. The key points of the concept are:

- plausibility check of 3d object detector by object models as a-priori knowledge
- comparison of predictions with object models via matching score
- focus on lane change use case 2

4.18.2 Use cases

4.18.2.1 **Optical misalignment of Scala sensors**:
The reliability of LiDAR sensors is crucial for any kind of object detection. So far these sensors are mainly used for driving assistance on motorways. Therefore we focus here on the use case 2 (lane change).

4.18.2.2 **Object model matching**:
In the course of the project Valeo plans to restrict the object detection and the proposed idea of matching object models to cars only. Car objects are the most suitable for the initial proof of concept, since they are rigid and have a well defined shape, in contrast

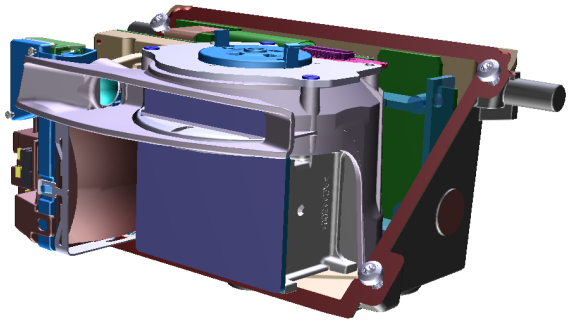


Fig. 58: Schematics of a Valeo Scala Sensor.

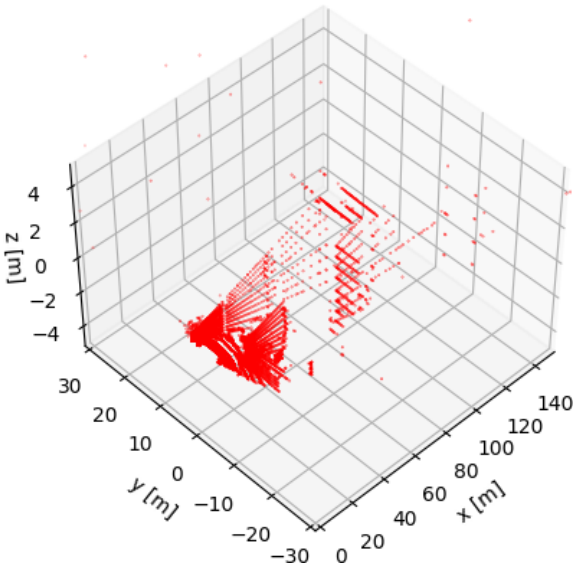


Fig. 59: A point cloud generated with a Valeo LiDAR sensor (model *Scala 2*). The scanning angle lies in the xy plane.

to for example pedestrians. Thus, the focus of the planned contributions is set to use case 2, where a challenging lane change is performed. During such a lane change reliable predictions of objects are crucial since it is time critical and for example a false positive detection might lead to an emergency break.

4.18.3 Knowledge sources

4.18.3.1 **Optical misalignment of Scala sensors**:
The LiDAR sensors from Valeo's product line *Scala* determine the distance of objects by measuring the time of flight of laser beams, which are reflected by an object. Currently, the most commonly used LiDAR in the automotive industry is the *mechanical spinning* LiDAR, where a mirror rotating around a vertical axis is used and several vertically aligned laser diodes are used to scan a certain solid angle [128]. The stack of laser diodes can additonally be multiplied in vertical direction to increase the field of view further. Figure 58 shows a schematic drawing of a *Scala* sensor. The result is a point cloud representing the environment of the car. Figure 59 is an example for such a point cloud. Up to now the *Scala* is the only automotive LiDAR sensor in series production [129].

The sensor degrades if the optical parts are misaligned with respect to each other for the reasons mentioned above. It is known that these misalignments are permanent and irreversible (expert knowledge).

4.18.3.2 **Object model matching**:
Valeo identified two possible sources for car models. First, object models could be taken from the public ShapeNet dataset [110]. The Shapenet dataset consists of CAD models of a variety of objects among others cars. These are synthetically created and thus the car models are not as one would expect on German roads. For example some have rear spoilers. Therefore, the objects may need to be manually curated. The other option is to extract object models from real world datasets such as nuScenes [130] or Kitti [131]. This method is greater effort but may lead to more realistic object models.

4.18.4 Representation/Formalization concept

4.18.4.1 **Optical misalignment of Scala sensors**:
The knowledge about permanence and irreversibility of the optical misalignment can be formalized by a monotonicity paramter in a, e. g., YAML file, which is set to true, meaning that misalignment is a monotonically increasing function.

If M_t is the misalignment at time t, the knowledge can be represented as the logical rule

$$M_{t_2} \geq M_{t_1} \tag{9}$$

for $t_2 > t_1$.

A further source of knowledge is the number of layers of the LiDAR. The Scala 2 sensor has, e. g., 16 layers, Scala 1 has only four layers, Velodyne produces LiDARs with 16, 32, 64 and 128 layers.

Another ageing effect is the failure of a whole photodiode stack. Its intended consequences are mentioned in the next section.

4.18.4.2 **Object model matching**:
For the representation of object models Valeo considers three different methods. First, object models could be represented as mesh grids. Secondly, a representation as dense point cloud is possible. The third option Valeo takes into account are gaussian mixture models for representation of car models. Here, for each relative angle of sensor and object a gaussian mixture model is learned from real world data scans.

4.18.5 Conformity check

4.18.5.1 **Optical misalignment of Scala sensors**:
The network outputs one of four possible misalignment classes (no/light/medium/strong misalignment). A plausibility check of the classification result is possible if the sensor saves the recognized misalignment classes over the time and compares it with earlier detected misalignment classes. Since misalignment is irreversible, it may not happen that a lighter misalignment is detected than before.

Furthermore the probability for the misalignment classes are available. If the probability for the detected class is not remarkably higher than those for the other classes the result has to be viewed critically.

Moreover, completely invalid point clouds can be fed into the network, for example empty clouds with no points or clouds with all three echos to all points which may happen in reality due to electrical or electronic fault. If a whole stack of photodiodes (four in the case of *Scala 2*) does not register any echo, this is a sign that one of the emitting laser diodes is damaged. In such a case the network is expected to output no high probability for a certain misalignment class.

4.18.5.2 **Object model matching**:
Valeo aims to use the predicted bounding boxes of a 3D object detector to cut out the points belonging to single objects and compare them to a selection of appropriate object models. The matching score of this comparison shall be used as indicator to discard or keep the predicted bounding box. Depending on the representation of the object models different matching metrics can be used. They range from the simple mean square error or distance metrics such as chamfer or earthmover distance to the usage of a likelihood. The number of objects that will be used for the comparison is in principle arbitrary for the conformity check. Nevertheless, Valeo aims to use only a small amount of different objects to reduce the complexity for the first proof of concept. For this proof of concept PointPillars [132] will be utilized as first 3D object detector in accordance with WP 1.1.

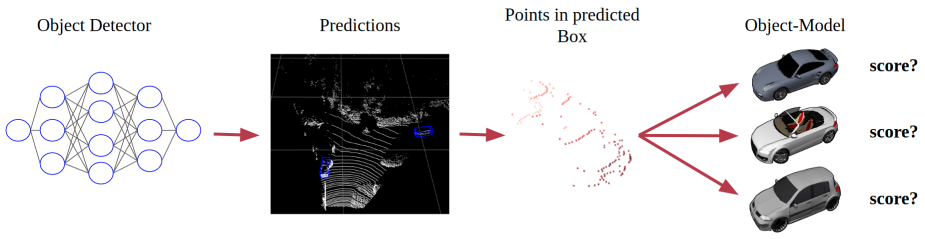


Fig. 60: Valeo: Illustration of planned contribution. The points inside the predicted bounding boxes of an 3d object detector are compared with a-priori object models. The matching score is used for the plausibility check.

5 Conclusion & Outlook

Within the present first iteration of deliverables D1 and D4 of the research project KI Wissen, the first methods and concepts for knowledge formalization, knowledge representation and conformance checking were presented. In addition, various knowledge sources were presented on which the concepts and methods are based. These methods are used in the context of KI Wissen to integrate prior knowledge into the training of an AI component, which is supposed to act in an automated vehicle with automation level between 3 to 5. The aim of this approach is to increase the generalization of the AI component, to reduce costs (e.g., data collection, data processing, data labeling), and to increase the efficiency (runtime and performance) of the training process itself. Prior knowledge considered here is both scientific knowledge (e.g. physics and mathematics) and world and expert knowledge (e.g. maps, traffic rules, social norms).

Based on three main use cases dealing with pedestrian perception (UC1), conscious, controlled rule exception (UC3) and planning for a lane change (UC2) (more details in 3.2), different concepts and methods were presented by a total of 16 project partners. The 16 project partners work on different fundamental knowledge sources. Here, the categories knowledge on evolution, knowledge on scene, traffic rules & requirements, spatial knowledge and social norms were identified. Within the partner contributions for knowledge formalization and knowledge representation, the categories equations & inequalities, logic, knowledge graphs, intermediate network representation, visual representation and mesh grid & point cloud were identified. Finally, for the concepts and methods of conformity checking of prior knowledge, the categories verifying semantic detection, monitoring, causal queries & sensitivity analysis, anomaly detection, conformity metrics and equivariance & likelihoods were identified. In total, more than 50 different concepts were presented and even more methods were addressed (more details in 4).

This document, is the first iteration of deliverables D1 and D4 in the KI Wissen project. The second iteration, which is due at the end of the 2nd project year (end of 2022), will present and evaluate the first formalized knowledge modules, as well as explain the first test methods for the knowledge conformity of the developed modules. Furthermore, contents from subproject 2 (SP2) will be explained, which develop concepts for the extraction of integrated prior knowledge in order to evaluate the quality of the integrated knowledge with selected metrics. Furthermore, additional concepts and methods from subproject 3 (SP3) will be shared, which deals with the identification and classification of knowledge that can be combined with data-driven AI approaches.

REFERENCES

[1] Bernd Becker et al. "SFB/TR 14 AVACS – Automatic Verification and Analysis of Complex Systems (Der Sonderforschungsbereich/Transregio 14 AVACS – Automatische Verifikation und Analyse komplexer Systeme)". In: *it - Information Technology* 49 (2007), pp. 118–126.

[2] Laura Von Rueden et al. "Informed machine learning–towards a taxonomy of explicit integration of knowledge into machine learning". In: *learning* 18 (2019), pp. 19–20.

[3] Laura Von Rueden et al. "Informed Machine Learning - A Taxonomy and Survey of Integrating Prior Knowledge into Learning Systems". In: *IEEE Transactions on Knowledge and Data Engineering* (2021). DOI: 10.1109/TKDE.2021.3079836.

[4] Nicolas Papernot and Patrick McDaniel. "Deep k-nearest neighbors: Towards confident, interpretable and robust deep learning". In: *arXiv preprint arXiv:1803.04765* (2018). URL: http://arxiv.org/abs/1803.04765.

[5] Jinhan Kim, Robert Feldt, and Shin Yoo. "Guiding Deep Learning System Testing using Surprise Adequacy". In: *CoRR* abs/1808.08444 (2018). arXiv: 1808.08444. URL: http://arxiv.org/abs/1808.08444.

[6] Wei Ma et al. "Test Selection for Deep Learning Systems". In: *CoRR* abs/1904.13195 (2019). arXiv: 1904.13195. URL: http://arxiv.org/abs/1904.13195.

[7] Yutao Han, Rina Tse, and Mark E. Campbell. "Pedestrian Motion Model Using Non-Parametric Trajectory Clustering and Discrete Transition Points". In: *CoRR* abs/2001.10571 (2020). arXiv: 2001.10571. URL: https://arxiv.org/abs/2001.10571.

[8] Heinz Hautzinger, Manfred Pfeiffer, and Jochen Schmidt. "Expansion of GIDAS Sample Data to the Regional Level: statistical Methodology and Practical Experiences". en. In: 2005, pp. 38–43. URL: https://bast.opus.hbz-nrw.de/frontdoor/index/index/docId/420 (visited on 10/29/2021).

[9] Dietmar Otte. *Unfallaufnahme bei der Polizei und neue Techniken (3D) - Erfahrungen aus Erhebungen am Unfallort Hannover und der Zusammenarbeit mit der Polizei*. Hannover, 2015. URL: https://docplayer.org/26086824-Gidas-german-in-depth-accident-study.html (visited on 10/29/2021).

[10] Claus Pastor. *GIDAS: den Verkehrsunfällen auf der Spur*. Dec. 2017. URL: https://www.bast.de/BASt_2017/EN/Automotive_Engineering/Subjects/info-gidas.pdf?__blob=publicationFile&v=3 (visited on 10/29/2021).

[11] WP29. *Framework document on automated/autonomous vehicles*. en. Mar. 2019. URL: https://globalautoregs.com/documents/18988 (visited on 10/29/2021).

[12] *Functional Requirements for Automated and Autonomous Vehicles (FRAV) - Transport - Vehicle Regulations - UNECE Wiki*. Sept. 2021. URL: https://wiki.unece.org/pages/viewpage.action?pageId=87622236 (visited on 10/29/2021).

[13] *Validation Method for Automated Driving (VMAD) - Transport - Vehicle Regulations - UNECE Wiki*. Oct. 2021. URL: https://wiki.unece.org/pages/viewpage.action?pageId=60361611 (visited on 10/29/2021).

[14] Forschungsgesellschaft für Straßen- und Verkehrswesen, ed. *Richtlinien für die Markierung von Straßen. Teil A: Autobahnen*. ger. Ausgabe 2019. FGSV 330A. Köln: Forschungsgesellschaft für Straßen- und Verkehrswesen e.V, 2019. ISBN: 978-3-86446-251-1.

[15] *Allgemeine Verwaltungsvorschrift zur Straßenverkehrs-Ordnung (VwV-StVO)*. Jan. 1999. URL: http://www.verwaltungsvorschriften-im-internet.de/bsvwvbund_26012001_S3236420014.htm (visited on 10/29/2021).

[16] Michael Färber, Boulos El Asmar, and Syrine Chelly. "AWARE: An Ontology for Situational Awareness of Autonomous Vehicles in Manufacturing". In: *Proceedings of the 2021 Commonsense Knowledge Graph Workshop (CSKG'21)@AAAI'21*. AAAI, 2021.

[17] Ji Eun Kim et al. "Accelerating Road Sign Ground Truth Construction with Knowledge Graph and Machine Learning". In: *arXiv:2012.02672 [cs]* (Dec. 2020). arXiv: 2012.02672. URL: http://arxiv.org/abs/2012.02672 (visited on 10/29/2021).

[18] Robert Azencott et al. *Analysis Methods for Accident Causation Studies*. Dec. 2007.

[19] Karsten Scheibler et al. "Solving Constraint Systems from Traffic Scenarios for the Validation of Autonomous Driving". In: SC^2. 2019.

[20] Kaiming He et al. "Mask R-CNN". In: *CoRR* abs/1703.06870 (2017). arXiv: 1703.06870. URL: http://arxiv.org/abs/1703.06870.

[21] Ross B. Girshick. "Fast R-CNN". In: *CoRR* abs/1504.08083 (2015). arXiv: 1504.08083. URL: http://arxiv.org/abs/1504.08083.

[22] Shaoqing Ren et al. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks". In: *CoRR* abs/1506.01497 (2015). arXiv: 1506.01497. URL: http://arxiv.org/abs/1506.01497.

[23] Alexey Dosovitskiy et al. "CARLA: An Open Urban Driving Simulator". In: *Proceedings of the 1st Annual Conference on Robot Learning*. 2017, pp. 1–16.

[24] OpenStreetMap contributors. *Planet dump retrieved from https://planet.osm.org*. https://www.openstreetmap.org. 2017.

[25] Laura von Rueden et al. "Street-Map Based Validation of Semantic Segmentation in Autonomous Driving". In: *2020 25th International Conference on Pattern Recognition (ICPR)*. IEEE. 2021, pp. 10203–10210.

[26] Ashish Vaswani et al. "Attention Is All You Need". In: *arXiv:1706.03762 [cs]* (Dec. 2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762 (visited on 10/25/2021).

[27] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding". In: *arXiv:1810.04805 [cs]* (May 2019). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805 (visited on 10/25/2021).

[28] Yinhan Liu et al. "RoBERTa: A Robustly Optimized BERT Pretraining Approach". In: *arXiv:1907.11692 [cs]* (July 2019). arXiv: 1907.11692. URL: http://arxiv.org/abs/1907.11692 (visited on 10/25/2021).

[29] Victor Sanh et al. "DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter". In: *arXiv:1910.01108 [cs]* (Feb. 2020). arXiv: 1910.01108. URL: http://arxiv.org/abs/1910.01108 (visited on 10/25/2021).

[30] Zhenzhong Lan et al. "ALBERT: A Lite BERT for Self-supervised Learning of Language Representations". In: *arXiv:1909.11942 [cs]* (Feb. 2020). arXiv: 1909.11942. URL: http://arxiv.org/abs/1909.11942 (visited on 10/25/2021).

[31] Zhilin Yang et al. "XLNet: Generalized Autoregressive Pretraining for Language Understanding". In: *arXiv:1906.08237 [cs]* (Jan. 2020). arXiv: 1906.08237. URL: http://arxiv.org/abs/1906.08237 (visited on 10/25/2021).

[32] Xiaozhi Wang et al. "KEPLER: A Unified Model for Knowledge Embedding and Pre-trained Language Representation". In: *arXiv:1911.06136 [cs]* (Nov. 2020). arXiv: 1911.06136. URL: http://arxiv.org/abs/1911.06136 (visited on 10/25/2021).

[33] Ruize Wang et al. "K-Adapter: Infusing Knowledge into Pre-Trained Models with Adapters". In: *arXiv:2002.01808 [cs]* (Dec. 2020). arXiv: 2002.01808. URL: http://arxiv.org/abs/2002.01808 (visited on 10/25/2021).

[34] Tianxiang Sun et al. "CoLAKE: Contextualized Language and Knowledge Embedding". In: *arXiv:2010.00309 [cs]* (Oct. 2020). arXiv: 2010.00309. URL: http://arxiv.org/abs/2010.00309 (visited on 10/25/2021).

[35] Ningyu Zhang et al. "Drop Redundant, Shrink Irrelevant: Selective Knowledge Injection for Language Pretraining". In: *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*. Ed. by Zhi-Hua Zhou. International Joint Conferences on Artificial Intelligence Organization, Aug. 2021, pp. 4007–4014. DOI: 10.24963/ijcai.2021/552. URL: https://doi.org/10.24963/ijcai.2021/552.

[36] Sebastian Reich and Colin Cotter. *Probabilistic forecasting and Bayesian data assimi-*

lation. Cambridge: Cambridge University Press, 2015. ISBN: 978-1-107-06939-8 978-1-107-66391-6.

[37] Cosma Rohilla Shalizi and James P. Crutchfield. "Computational mechanics: Pattern and prediction, structure and simplicity". In: *Journal of Statistical Physics* 104.3/4 (2001), pp. 817–879. ISSN: 00224715. DOI: 10.1023/A:1010388907793. URL: http://link.springer.com/10.1023/A:1010388907793 (visited on 11/25/2021).

[38] Simon S. Haykin, ed. *Kalman filtering and neural networks*. Adaptive and learning systems for signal processing, communications, and control. New York: Wiley, 2001. ISBN: 978-0-471-36998-1.

[39] Greg Welch, Gary Bishop, et al. "An introduction to the Kalman filter". In: (1995). Publisher: Chapel Hill, NC, USA.

[40] Jun S. Liu and Rong Chen. "Sequential Monte Carlo Methods for Dynamic Systems". en. In: *Journal of the American Statistical Association* 93.443 (Sept. 1998), pp. 1032–1044. ISSN: 0162-1459, 1537-274X. DOI: 10.1080/01621459.1998.10473765. URL: http://www.tandfonline.com/doi/full/10.1080/01621459.1998.10473765 (visited on 11/25/2021).

[41] Dieter Schramm, Roberto Bardini, and Manfred Hiller. *Vehicle Dynamics: Modeling and Simulation*. 2nd ed. 2018. Berlin, Heidelberg: Springer Berlin Heidelberg : Imprint: Springer, 2018. ISBN: 978-3-662-54483-9. DOI: 10.1007/978-3-662-54483-9.

[42] Tobias Gindele, Sebastian Brechtel, and Rudiger Dillmann. "A probabilistic model for estimating driver behaviors and vehicle trajectories in traffic environments". In: *13th International IEEE Conference on Intelligent Transportation Systems*. Funchal, Madeira Island, Portugal: IEEE, Sept. 2010, pp. 1625–1631. ISBN: 978-1-4244-7657-2. DOI: 10.1109/ITSC.2010.5625262. URL: http://ieeexplore.ieee.org/document/5625262/ (visited on 11/03/2021).

[43] Matthias Schreier, Volker Willert, and Jurgen Adamy. "An Integrated Approach to Maneuver-Based Trajectory Prediction and Criticality Assessment in Arbitrary Road Environments". In: *IEEE Transactions on Intelligent Transportation Systems* 17.10 (Oct. 2016), pp. 2751–2766. ISSN: 1524-9050, 1558-0016. DOI: 10.1109/TITS.2016.2522507. URL: http://ieeexplore.ieee.org/document/7412746/ (visited on 11/05/2021).

[44] Chanho Kim et al. "Multiple Hypothesis Tracking Revisited". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Dec. 2015.

[45] Angel F. Garcia-Fernandez et al. "Poisson Multi-Bernoulli Mixture Filter: Direct Derivation and Implementation". In: *IEEE Transactions on Aerospace and Electronic Systems* 54.4 (Aug. 2018), pp. 1883–1901. ISSN: 0018-9251, 1557-9603, 2371-9877. DOI: 10.1109/TAES.2018.2805153. URL: https://ieeexplore.ieee.org/document/8289337/ (visited on 11/05/2021).

[46] Karl Granström et al. "Poisson Multi-Bernoulli Mixtures for Sets of Trajectories". In: *arXiv:1912.08718 [cs, eess, stat]* (Dec. 2019). arXiv: 1912.08718. URL: http://arxiv.org/abs/1912.08718 (visited on 11/05/2021).

[47] Charles C. Macadam. "Understanding and Modeling the Human Driver". In: *Vehicle System Dynamics* 40.1-3 (Jan. 2003), pp. 101–134. ISSN: 0042-3114. DOI: 10.1076/vesd.40.1.101.15875. URL: http://www.tandfonline.com/doi/abs/10.1076/vesd.40.1.101.15875 (visited on 11/25/2021).

[48] I. Bae et al. "Self-Driving like a Human driver instead of a Robocar: Personalized comfortable driving experience for autonomous vehicles". In: *arXiv:2001.03908 [cs, eess]* (Jan. 2020). arXiv: 2001.03908. URL: http://arxiv.org/abs/2001.03908 (visited on 11/25/2021).

[49] Hasti Hayati et al. "Jerk within the Context of Science and Engineering—A Systematic Review". en. In: *Vibration* 3.4 (Oct. 2020), pp. 371–409. ISSN: 2571-631X. DOI: 10.3390/vibration3040025. URL: https://www.mdpi.com/2571-631X/3/4/25 (visited on 11/25/2021).

[50] Francesco de Dilectis, Daniele Mortari, and Renato Zanetti. "Bézier Description of Space

Trajectories". en. In: *Journal of Guidance, Control, and Dynamics* 39.11 (Nov. 2016), pp. 2535–2539. ISSN: 0731-5090, 1533-3884. DOI: 10.2514/1.G000719. URL: https://arc.aiaa.org/doi/10.2514/1.G000719 (visited on 11/25/2021).

[51] Julian J. Faraway, Matthew P. Reed, and Jing Wang. "Modelling three-dimensional trajectories by using Bézier curves with application to hand motion: *Modelling Three-dimensional Trajectories*". en. In: *Journal of the Royal Statistical Society: Series C (Applied Statistics)* 56.5 (Nov. 2007), pp. 571–585. ISSN: 00359254. DOI: 10.1111/j.1467-9876.2007.00592.x. URL: https://onlinelibrary.wiley.com/doi/10.1111/j.1467-9876.2007.00592.x (visited on 11/25/2021).

[52] Ronny Hug, Wolfgang Hübner, and Michael Arens. "Modeling continuous-time stochastic processes using $\mathcal{N}$-Curve mixtures". In: *arXiv:1908.04030 [cs, stat]* (Sept. 2019). arXiv: 1908.04030. URL: http://arxiv.org/abs/1908.04030 (visited on 11/25/2021).

[53] Moritz Werling et al. "Optimal trajectory generation for dynamic street scenarios in a Frenét Frame". In: *2010 IEEE International Conference on Robotics and Automation*. Anchorage, AK: IEEE, May 2010, pp. 987–993. ISBN: 978-1-4244-5038-1. DOI: 10.1109/ROBOT.2010.5509799. URL: http://ieeexplore.ieee.org/document/5509799/ (visited on 11/25/2021).

[54] Shashwati Ray and PSV Nataraj. "A Matrix Method for Efficient Computation of Bernstein Coefficients." In: *Reliab. Comput.* 17.1 (2012), pp. 40–71.

[55] M. Yamakado and M. Abe. "An experimentally confirmed driver longitudinal acceleration control model combined with vehicle lateral motion". en. In: *Vehicle System Dynamics* 46.sup1 (Sept. 2008), pp. 129–149. ISSN: 0042-3114, 1744-5159. DOI: 10.1080/00423110701882363. URL: http://www.tandfonline.com/doi/abs/10.1080/00423110701882363 (visited on 11/25/2021).

[56] A. Takahashi et al. "Local Path Planning And Motion Control For Agv In Positioning". In: *Proceedings. IEEE/RSJ International Workshop on Intelligent Robots and Systems. (IROS '89) 'The Autonomous Mobile Robots and Its Applications*. Tsukuba, Japan: IEEE, 1989, pp. 392–397. DOI: 10.1109/IROS.1989.637936. URL: http://ieeexplore.ieee.org/document/637936/ (visited on 11/25/2021).

[57] Gustavo Arechavaleta et al. "An Optimality Principle Governing Human Walking". In: *IEEE Transactions on Robotics* 24.1 (Feb. 2008), pp. 5–14. ISSN: 1552-3098. DOI: 10.1109/TRO.2008.915449. URL: http://ieeexplore.ieee.org/document/4456738/ (visited on 11/25/2021).

[58] Chenhan Jiang et al. "Hybrid knowledge routed modules for large-scale object detection". In: *arXiv preprint arXiv:1810.12681* (2018).

[59] Shaoqing Ren et al. "Faster R-CNN: towards real-time object detection with region proposal networks". In: *IEEE transactions on pattern analysis and machine intelligence* 39.6 (2016), pp. 1137–1149.

[60] Judea Pearl. *Causality: models, reasoning, and inference*. en. 2. ed., repr. with corr. Cambridge: Cambridge University Press, 2013. ISBN: 978-0-521-77362-1 978-0-521-89560-6.

[61] Manfred Mitschke and Henning Wallentowitz. *Dynamik der Kraftfahrzeuge*. de. Wiesbaden: Springer Fachmedien Wiesbaden, 2014. ISBN: 978-3-658-05067-2 978-3-658-05068-9. DOI: 10.1007/978-3-658-05068-9. URL: http://link.springer.com/10.1007/978-3-658-05068-9.

[62] Julius Ziegler. "Optimale Bahn- und Trajektorienplanung für Automobile". de. In: (2015). Medium: PDF Publisher: Karlsruhe. DOI: 10.5445/IR/1000057846. URL: http://digbib.ubka.uni-karlsruhe.de/volltexte/1000057846.

[63] Ramalingam Shanmugam. "Elements of causal inference: foundations and learning algorithms". en. In: *Journal of Statistical Computation and Simulation* 88.16 (Nov. 2018). Number: 16, pp. 3248–3248. ISSN: 0094-9655, 1563-5163. DOI: 10.1080/00949655.2018.1505197. URL: https://www.tandfonline.com/doi/full/10.1080/00949655.2018.1505197.

[64] Judea Pearl. "The seven tools of causal inference, with reflections on machine learning". en. In: *Communications of the ACM* 62.3 (Feb. 2019). Number: 3, pp. 54–60. ISSN: 00010782. DOI: 10.1145/3241036. URL: http://dl.acm.org/citation.cfm?doid=3314328.3241036.

[65] James M. Robins, Miguel Ángel Hernán, and Babette Brumback. "Marginal Structural Models and Causal Inference in Epidemiology:" en. In: *Epidemiology* 11.5 (Sept. 2000), pp. 550–560. ISSN: 1044-3983. DOI: 10.1097/00001648-200009000-00011. URL: http://journals.lww.com/00001648-200009000-00011.

[66] Stephen L. Morgan and Christopher Winship. *Counterfactuals and causal inference: methods and principles for social research*. eng. Second edition, reprinted with corrections. Analytical methods for social research. Cambridge: Cambridge University Press, 2015. ISBN: 978-1-107-06507-9 978-1-107-69416-3.

[67] Paul Spannaus, Peter Zechel, and Kilian Lenz. "AUTOMATUM DATA: Drone-based highway dataset for the development and validation of automated driving software for research and commercial applications". In: *2021 32st IEEE Intelligent Vehicles Symposium (IV)*. 2021.

[68] Nick Pawlowski, Daniel C. Castro, and Ben Glocker. "Deep Structural Causal Models for Tractable Counterfactual Inference". In: *arXiv:2006.06485 [cs, stat]* (Oct. 2020). arXiv: 2006.06485. URL: http://arxiv.org/abs/2006.06485.

[69] Lars Buesing et al. "Woulda, Coulda, Shoulda: Counterfactually-Guided Policy Search". In: *arXiv:1811.06272 [cs, stat]* (Nov. 2018). arXiv: 1811.06272 version: 1. URL: http://arxiv.org/abs/1811.06272.

[70] Julian Bernhard et al. "BARK: Open behavior benchmarking in multi-agent environments". In: IEEE, 2020, pp. 6201–6208. DOI: 10.1109/CAVS51000.2020.9334599.

[71] Klemens Esterle, Luis Gressenbuch, and Alois Knoll. "Formalizing Traffic Rules for Machine Interpretability". In: IEEE, Nov. 2020, pp. 1–7. ISBN: 978-1-7281-9001-3. DOI: 10.1109/CAVS51000.2020.9334599.

[72] Giovanni Sartor. *Legal Reasoning: A Cognitive Approach to Law*. Springer, Jan. 2005. ISBN: 978-1-4020-3387-2.

[73] Tara Athan et al. "LegalRuleML: Design Principles and Foundations". en. In: *Reasoning Web. Web Logic Rules: 11th International Summer School 2015, Berlin, Germany, July 31-August 4, 2015, Tutorial Lectures.* Ed. by Wolfgang Faber and Adrian Paschke. Lecture Notes in Computer Science. Cham: Springer International Publishing, 2015, pp. 151–188. ISBN: 978-3-319-21768-0. DOI: 10.1007/978-3-319-21768-0_6. URL: https://doi.org/10.1007/978-3-319-21768-0_6 (visited on 03/19/2021).

[74] Ho-Pun Lam, Mustafa Hashmi, and Brendan Scofield. *Enabling Reasoning with LegalRuleML*. Vol. 9718. Pages: 257. July 2016. ISBN: 978-3-319-42018-9. DOI: 10.1007/978-3-319-42019-6_16.

[75] Mehdi Rohaninezhad, Shereena Mohd Arif, and Shahrul Azman Mohd Noah. "A grounder for SPINdle defeasible logic reasoner". en. In: *Expert Systems with Applications* 42.20 (Nov. 2015), pp. 7098–7109. ISSN: 09574174. DOI: 10.1016/j.eswa.2015.04.065. URL: https://linkinghub.elsevier.com/retrieve/pii/S0957417415003073 (visited on 11/02/2021).

[76] Luciano Serafini and Artur d'Avila Garcez. "Logic Tensor Networks: Deep Learning and Logical Reasoning from Data and Knowledge". In: *arXiv:1606.04422 [cs]* (July 2016). arXiv: 1606.04422. URL: http://arxiv.org/abs/1606.04422 (visited on 12/22/2020).

[77] Livio Robaldo et al. "Formalizing GDPR Provisions in Reified I/O Logic: The DAPRECO Knowledge Base". en. In: *Journal of Logic, Language and Information* 29.4 (Dec. 2020), pp. 401–449. ISSN: 0925-8531, 1572-9583. DOI: 10.1007/s10849-019-09309-z. URL: http://link.springer.com/10.1007/s10849-019-09309-z (visited on 03/19/2021).

[78] Monica Palmirani and Fabio Vitali. *OASIS LegalDocumentML (LegalDocML) TC*. 2021. URL: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=legaldocml.

[79] Mark Campbell et al. "Autonomous driving in urban environments: approaches, lessons and challenges". In: *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 368.1928 (2010).

[80] Scott Drew Pendleton et al. "Perception, planning, control, and coordination for autonomous vehicles". In: *Machines* 5.1 (2017).

[81] Marius Cordts et al. "The cityscapes dataset for semantic urban scene understanding". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2016.

[82] Shanshan Zhang, Rodrigo Benenson, and Bernt Schiele. "Citypersons: A diverse dataset for pedestrian detection". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017, pp. 3213–3221.

[83] Eduardo Romera et al. "Erfnet: Efficient residual factorized convnet for real-time semantic segmentation". In: *IEEE Transactions on Intelligent Transportation Systems* 19.1 (2017), pp. 263–272.

[84] Brian Paden et al. "A Survey of Motion Planning and Control Techniques for Self-driving Urban Vehicles". en. In: *arXiv:1604.07446 [cs]* (Apr. 2016). arXiv: 1604.07446. URL: http://arxiv.org/abs/1604.07446 (visited on 11/29/2020).

[85] Wenyuan Zeng et al. "End-to-end Interpretable Neural Motion Planner". In: *arXiv:2101.06679 [cs]* (Jan. 2021). arXiv: 2101.06679. URL: http://arxiv.org/abs/2101.06679 (visited on 02/24/2021).

[86] BSI. *Operational Design Domain (ODD) taxonomy for an automated driving system (ADS) - Specification*. ISBN 978 0 539 06735 4. Aug. 2020. URL: https://www.bsigroup.com/globalassets/localfiles/en-gb/cav/pas1883.pdf (visited on 06/20/2021).

[87] N. Aréchiga. "Specifying Safety of Autonomous Vehicles in Signal Temporal Logic". In: *2019 IEEE Intelligent Vehicles Symposium (IV)*. ISSN: 2642-7214. June 2019, pp. 58–63. DOI: 10.1109/IVS.2019.8813875.

[88] Klemens Esterle, Luis Gressenbuch, and Alois Knoll. "Formalizing Traffic Rules for Machine Interpretability". In: *arXiv:2007.00330 [cs]* (Sept. 2020). arXiv: 2007.00330. URL: http://arxiv.org/abs/2007.00330 (visited on 01/11/2021).

[89] S. Maierhofer et al. "Formalization of Interstate Traffic Rules in Temporal Logic". In: *2020 IEEE Intelligent Vehicles Symposium (IV)*. ISSN: 2642-7214. Oct. 2020, pp. 752–759. DOI: 10.1109/IV47402.2020.9304549.

[90] Martin Buechel et al. "Ontology-based traffic scene modeling, traffic regulations dependent situational awareness and decision-making for automated vehicles". In: *2017 IEEE Intelligent Vehicles Symposium (IV)*. June 2017, pp. 1471–1476. DOI: 10.1109/IVS.2017.7995917.

[91] Yafu Tian et al. "Road Scene Graph: A Semantic Graph-Based Scene Representation Dataset for Intelligent Vehicles". In: *arXiv:2011.13588 [cs]* (Nov. 2020). arXiv: 2011.13588. URL: http://arxiv.org/abs/2011.13588 (visited on 04/07/2021).

[92] Mayank Bansal, Alex Krizhevsky, and Abhijit Ogale. "ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst". In: *arXiv:1812.03079 [cs]* (Dec. 2018). arXiv: 1812.03079. URL: http://arxiv.org/abs/1812.03079 (visited on 01/21/2021).

[93] Sergio Casas et al. "The Importance of Prior Knowledge in Precise Multimodal Prediction". In: *arXiv:2006.02636 [cs, stat]* (June 2020). arXiv: 2006.02636. URL: http://arxiv.org/abs/2006.02636 (visited on 06/03/2021).

[94] Wenyuan Zeng et al. "DSDNet: Deep Structured self-Driving Network". In: *arXiv:2008.06041 [cs]* (Aug. 2020). arXiv: 2008.06041. URL: http://arxiv.org/abs/2008.06041 (visited on 02/01/2021).

[95] Jerry Liu et al. "Deep Structured Reactive Planning". In: *arXiv:2101.06832 [cs]* (Jan. 2021). arXiv: 2101.06832. URL: http://arxiv.org/abs/2101.06832 (visited on 02/07/2021).

[96] Diederik P. Kingma and Max Welling. "Auto-Encoding Variational Bayes". In: *arXiv:1312.6114 [cs, stat]* (May 2014). arXiv: 1312.6114. URL: http://arxiv.org/abs/1312.6114 (visited on 06/22/2021).

[97] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. "On a Formal Model of Safe and Scalable Self-driving Cars". In: *arXiv:1708.06374 [cs, stat]* (Oct. 2018). arXiv: 1708.06374. URL: http://arxiv.org/abs/1708.06374 (visited on 01/11/2021).

[98] David Nistér et al. "The safety force field". In: *NVIDIA White Paper* (2019).

[99] Klemens Esterle, Luis Gressenbuch, and Alois Knoll. "Modeling and Testing Multi-Agent Traffic Rules within Interactive Behavior Planning". In: *arXiv:2009.14186 [cs]* (Sept. 2020). arXiv: 2009.14186. URL: http://arxiv.org/abs/2009.14186 (visited on 03/01/2021).

[100] Lorenz Wellhausen and Mithun George Jacob. "Map-optimized probabilistic traffic rule evaluation". In: *2016 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. ISSN: 2153-0866. Oct. 2016, pp. 3012–3017. DOI: 10.1109/IROS.2016.7759466.

[101] Abbas Sadat et al. "Perceive, Predict, and Plan: Safe Motion Planning Through Interpretable Semantic Representations". en. In: *Computer Vision – ECCV 2020*. Ed. by Andrea Vedaldi et al. Vol. 12368. Series Title: Lecture Notes in Computer Science. Cham: Springer International Publishing, 2020, pp. 414–430. ISBN: 978-3-030-58591-4 978-3-030-58592-1. DOI: 10.1007/978-3-030-58592-1_25. URL: http://link.springer.com/10.1007/978-3-030-58592-1_25 (visited on 01/29/2021).

[102] Abbas Sadat et al. "Jointly Learnable Behavior and Trajectory Planning for Self-Driving Vehicles". In: *arXiv:1910.04586 [cs]* (Oct. 2019). arXiv: 1910.04586. URL: http://arxiv.org/abs/1910.04586 (visited on 02/01/2021).

[103] Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. "A reduction of imitation learning and structured prediction to no-regret online learning". In: *Proceedings of the fourteenth international conference on artificial intelligence and statistics*. JMLR Workshop and Conference Proceedings, 2011, pp. 627–635.

[104] Daniel S. Brown et al. "Value Alignment Verification". In: *arXiv:2012.01557 [cs]* (June 2021). arXiv: 2012.01557. URL: http://arxiv.org/abs/2012.01557 (visited on 06/30/2021).

[105] Yarin Gal. "Uncertainty in Deep Learning". In: *PhD Thesis* October (2016), p. 174. ISSN: 18736246. DOI: 10.1371/journal.pcbi.1005062. URL: http://www.cs.ox.ac.uk/people/yarin.gal/website/thesis/thesis.pdf.

[106] Alex H. Lang et al. "PointPillars: Fast Encoders for Object Detection from Point Clouds". In: (2018). arXiv: 1812.05784. URL: http://arxiv.org/abs/1812.05784.

[107] David Nistér et al. "The Safety Force Field". In: *NVIDIA White Paper* (2019). URL: https://www.nvidia.com/content/dam/en-zz/Solutions/self-driving-cars/safety-force-field/the-safety-force-field.pdf.

[108] Andrea Censi et al. "Liability, ethics, and culture-aware behavior specification using rulebooks". In: *Proceedings - IEEE International Conference on Robotics and Automation* 2019-May (2019), pp. 8536–8542. ISSN: 10504729. DOI: 10.1109/ICRA.2019.8794364. arXiv: 1902.09355.

[109] M Botsch et al. "OpenMesh – a generic and efficient polygon mesh data structure". In: (), p. 5.

[110] Angel X. Chang et al. *ShapeNet: An Information-Rich 3D Model Repository*. Tech. rep. arXiv:1512.03012 [cs.GR]. Stanford University — Princeton University — Toyota Technological Institute at Chicago, 2015.

[111] Stijn Oomes, Peter Snoeren, and Tjeerd Dijkstra. "3D shape representation: Transforming polygons into voxels". In: *Scale-Space Theory in Computer Vision*. Ed. by Bart ter Haar Romeny et al. Berlin, Heidelberg: Springer Berlin Heidelberg, 1997, pp. 349–352. ISBN: 978-3-540-69196-9.

[112] Jeong Joon Park et al. *DeepSDF: Learning Continuous Signed Distance Functions for Shape Representation*. 2019. arXiv: 1901.05103 [cs.CV].

[113] Francis Engelmann, Jörg Stückler, and Bastian Leibe. "Joint object pose estimation and shape reconstruction in urban street scenes using 3D shape priors". In: *Lecture Notes in Computer Science (including subseries Lecture*

Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) 9796 LNCS (2016), pp. 219–230. ISSN: 16113349. DOI: 10.1007/978-3-319-45886-1_18.

[114] Jürg Kohlas. "The mathematical theory of evidence — A short introduction". In: *System Modelling and Optimization: Proceedings of the Seventeenth IFIP TC7 Conference on System Modelling and Optimization, 1995*. Ed. by Jaroslav Doležal and Jiří Fidler. Boston, MA: Springer US, 1996, pp. 37–53. ISBN: 978-0-387-34897-1. DOI: 10.1007/978-0-387-34897-1_4. URL: https://doi.org/10.1007/978-0-387-34897-1_4.

[115] Geoffrey Hinton. "Products of Experts". In: 2007.

[116] Werner Damm et al. *Traffic Sequence Charts - From Visualization to Semantics*. Tech. rep. AVACS Technical Report 117 (Oct 2017), 2017.

[117] Ezio Bartocci et al. "Introduction to Runtime Verification". In: Feb. 2018, pp. 1–33. ISBN: 978-3-319-75631-8. DOI: 10.1007/978-3-319-75632-5_1.

[118] Oded Maler and D. Nickovic. "Monitoring Temporal Properties of Continuous Signals". In: *FORMATS/FTRTFT*. 2004.

[119] Andreas Bauer, Martin Leucker, and Christian Schallhart. "Runtime Verification for LTL and TLTL". In: 20.4 (2011). ISSN: 1049-331X. DOI: 10.1145/2000799.2000800. URL: https://doi.org/10.1145/2000799.2000800.

[120] Alexandre Donzé. "On Signal Temporal Logic". In: *Runtime Verification*. Ed. by Axel Legay and Saddek Bensalem. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013, pp. 382–383. ISBN: 978-3-642-40787-1.

[121] Dario Della Monica et al. "Interval Temporal Logics: a Journey". In: *Bulletin of the European Association for Theoretical Computer Science EATCS* 105 (Jan. 2011).

[122] Danièle Beauquier. "On probabilistic timed automata". In: *Theoretical Computer Science* 292.1 (2003), pp. 65–84. ISSN: 0304-3975. DOI: https://doi.org/10.1016/S0304-3975(01)00215-8. URL: https://www.sciencedirect.com/science/article/pii/S0304397501002158.

[123] Eleni Zapridou, Ezio Bartocci, and Panagiotis Katsaros. "Runtime Verification of Autonomous Driving Systems in CARLA". In: Oct. 2020. ISBN: 978-3-030-60507-0. DOI: 10.1007/978-3-030-60508-7_9.

[124] Michael Goedicke, Moritz Balz, and Michael Striewe. "UPPAAL-Modelle als ausführbare Spezifikation in Java." In: Jan. 2008, pp. 212–218.

[125] Wolfram Hardt Daniel Opp Mirko Caspar. "Code Generation for Timed Automata System Specifications Considering Target Platform Resources-Restrictions". In: Bangkok, Thailand, pp. 144–149.

[126] Paola Vallejo et al. "Towards a new template for the specification of requirements in semi-structured natural language". In: *Journal of Software Engineering Research and Development* 8 (Feb. 2020), p. 3. DOI: 10.5753/jserd.2020.473.

[127] Yulan Guo et al. *Deep Learning for 3D Point Clouds: A Survey*. 2020. arXiv: 1912.12033 [cs.CV]. URL: https://arxiv.org/abs/1912.12033.

[128] You Li and Javier Ibanez-Guzman. "Lidar for Autonomous Driving: The Principles, Challenges, and Trends for Automotive Lidar and Perception Systems". In: *IEEE Signal Processing Magazine* 37.4 (July 2020), pp. 50–61. ISSN: 1558-0792. DOI: 10.1109/MSP.2020.2973615. URL: https://ieeexplore.ieee.org/abstract/document/9127855.

[129] Joachim Mathes. 2021. URL: https://www.valeo.com/wp-content/uploads/2021/01/Evercore-CES-13-Jan-2021-Marc-Vrecko-%5C_-Joachim-Mathes.pdf.

[130] Holger Caesar et al. "nuScenes: A multimodal dataset for autonomous driving". In: *arXiv preprint arXiv:1903.11027* (2019).

[131] Andreas Geiger, Philip Lenz, and Raquel Urtasun. "Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite". In: *Conference on Computer Vision and Pattern Recognition (CVPR)*. 2012.

[132] Alex H. Lang et al. "PointPillars: Fast Encoders for Object Detection from Point Clouds". In: *CoRR* abs/1812.05784 (2018).

arXiv: 1812.05784. URL: http://arxiv.org/abs/1812.05784.