

What does AI need to know to drive: Testing relevance of knowledge

Dominik Grundt ^{ID,*}, Astrid Rakow, Philipp Borchers, Eike Möhlmann

German Aerospace Center e.V. - Institute of Systems Engineering for Future Mobility, Oldenburg, Germany

ARTICLE INFO

Keywords:

Knowledge-infused AI
Relevance
AI driving functions

ABSTRACT

Artificial Intelligence (AI) plays an important role in managing the complexity of automated driving. Nonetheless, training and ensuring the safety of AI is challenging. The safe generalization from a known to an unknown situation remains an unsolved problem. Infusing knowledge into AI driving functions seems a promising approach to address generalization, development costs, and training efficiency. We reason that ascertaining the relevance of infused knowledge provides a strong indication of the correct execution of previous development phases of knowledge infusion. As a causal reason for AI performance, relevant knowledge is important for explaining AI behavior. This paper defines a novel notion of *relevant knowledge* in knowledge-infused AI and for requirements satisfaction in traffic scenarios. We present a scenario-based testing procedure that not only checks whether a knowledge-infused AI model satisfies a given requirement R but also provides statements on the relevance of infused knowledge. Finally, we describe a systematic method for generating abstract knowledge scenarios to enable an efficient application of our relevance testing procedure.

1. Introduction

The development of highly automated driving functions for transportation is advancing rapidly [1]. The use of Artificial Intelligence (AI) modules for path planning [2], perception [3], and decision-making [4] promises to improve safety, efficiency, and comfort of mobility [5]. If automated mobility is to become established in public transport, AI driving functions must be safe, trustworthy, and socially acceptable. In order to meet these requirements, human drivers usually go through a driving school and have extensive experience from being exposed to traffic since early childhood, e.g., as passengers or pedestrians. They can draw on their acquired knowledge of traffic rules, physics, and social and societal norms to solve novel traffic scenarios.

Many AI approaches attempt to learn to drive based on a large amount of data. Unfortunately, this attempt is limited by the quantity and quality of the available data. Further, collecting real data and annotating the recorded data is tremendously time-consuming and costly. In recent years, the generation of suitable synthetic data has already significantly reduced the cost and time required for data-driven AI approaches [6]. However, AIs' generalization capabilities still need to be improved, and obtaining data for every possible situation seems intractable even with synthetic data. Hence, different approaches are needed.

Recently, there has been a focus in the AI community on using knowledge to develop AI driving functions [7]. One such application is *Knowledge Infusion*. Knowledge Infusion aims to bring prior knowledge (e.g., in transportation domain knowledge from physical laws, traffic rules, and social norms) into the AI. The resulting AI should generalize beyond the training data set, consider rules and social

* Corresponding author.

E-mail addresses: dominik.grundt@dlr.de (D. Grundt), astrid.rakow@dlr.de (A. Rakow), eike.moehlmann@dlr.de (E. Möhlmann).

<https://doi.org/10.1016/j.scico.2025.103297>

Received 14 June 2024; Received in revised form 17 January 2025; Accepted 27 February 2025

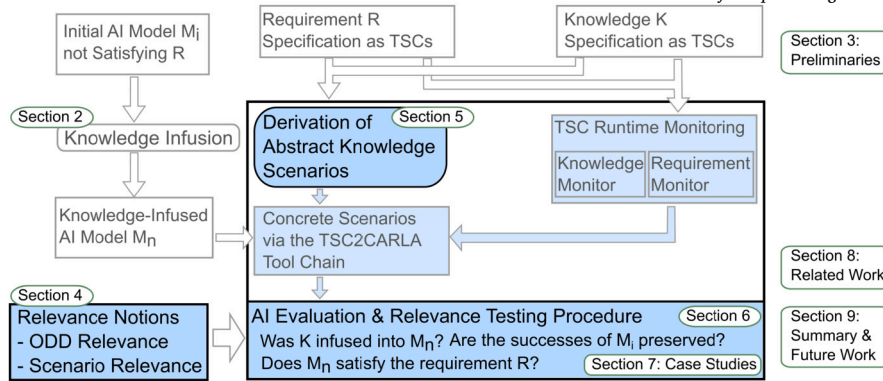


Fig. 1. Overview of our contribution. This paper focuses on the notions of relevance and our testing procedure to assess the relevance of infused knowledge and AI's performance.

norms, and act more safely in novel scenarios. Knowledge-infused AI thus seems to provide a means to yield more trustworthy [8] and social [9] AI models than purely black box data-driven approaches. There are various methods for knowledge infusion where either (1) knowledge is explicitly formalized or (2) indirectly given via enriched data sets containing diverse knowledge. For more details on the various techniques and their challenges, we refer to the surveys [10] and [11]. Knowledge-infused AI has already been successfully used in many domains: modeling physical dynamics [12], automated driving functions [13,14], medical prediction [15], manufacturing [16], and gaming agents [17]. A more detailed review of related work can be found Section 8.

In the context of knowledge infusion, we focus on the relevance of prior knowledge in the automotive domain. We consider prior knowledge that is specified in a symbolical logic and is either (i) a refinement of requirements or (ii) a description of critical scenarios. We consider an AI development process as described in [10] where knowledge is first *identified* by, e.g., domain experts, and then the knowledge is *formalized* before it gets *integrated* into the AI model. The process is described in more detail in Section 2. This process of knowledge-infusion is an active research field [10,18]. We address the following research question within this field:

RQ: Is the infused knowledge relevant for an AI model to fulfill its task?

Intuitively, knowledge is relevant if it needs to be infused into the AI because the AI does not satisfy its requirements otherwise. As AI training is costly, being able to provide relevant knowledge hence will enhance effective AI development. However, the above question *Q* is rarely considered in current development processes and the concept of *relevance* is not formally defined in this context. The work presented in this paper is driven by practical experience gained from the research project KI Wissen [7] and is further substantiated by the challenges identified in [19], which we discuss in Section 4.1.

Our contributions are threefold. For one, to answer the question *RQ*, we define a notion of *relevant knowledge* of knowledge-infused AI for requirements satisfaction in traffic scenarios (cf. Section 4). This notion captures the intuition that knowledge is relevant if it needs to be infused into the AI model to enable it to satisfy its requirements. We argue that a formal notion is the foundation for building a database of relevant domain knowledge for the development of new AI models targeting similar functionalities. Moreover, our notion is such that it captures a causal reason for a model's behavior and thus improves the explainability of AI models [20,21]. These benefits for the development of AI are discussed in more detail in Section 4.1.

Our second contribution is a testing procedure for relevance given a system requirement R and corresponding knowledge K . This procedure indicates whether the infused knowledge K is

- (a) valid, incomplete or incorrect formalized,
- (b) considered by the AI, and
- (c) relevant for an AI driving function to satisfy R .

The derived indicators (summarized in Table 5) make it possible to validate development phases, to diagnose issues of knowledge infusion and to control the storage of relevant knowledge. This method assumes a given AI model. As this testing procedure is scenario-based, it can be incorporated seamlessly into state-of-the-art scenario-based validation and verification of automated driving functions [22,23].

Our third contribution is a systematic method for deriving abstract knowledge scenarios, which enables us to execute our relevance testing procedure efficiently. Here, we exploit the visual yet formal specification language Traffic Sequence Charts (TSCs) [24] for scenario-based system requirements specification, corresponding knowledge, and abstract knowledge scenarios.

Fig. 1 gives an overview of our contributions within the context of our work. We only present the context as far as necessary and focus on the relevance notions and our testing procedure to assess the relevance of infused knowledge and the AI's performance.

Given requirements and knowledge formalizable as Traffic Sequence Charts (see [14] for a taxonomy of TSC formalizable knowledge and [25] for TSC formalizable requirements) and an AI model, our method assesses the relevance of infused knowledge.

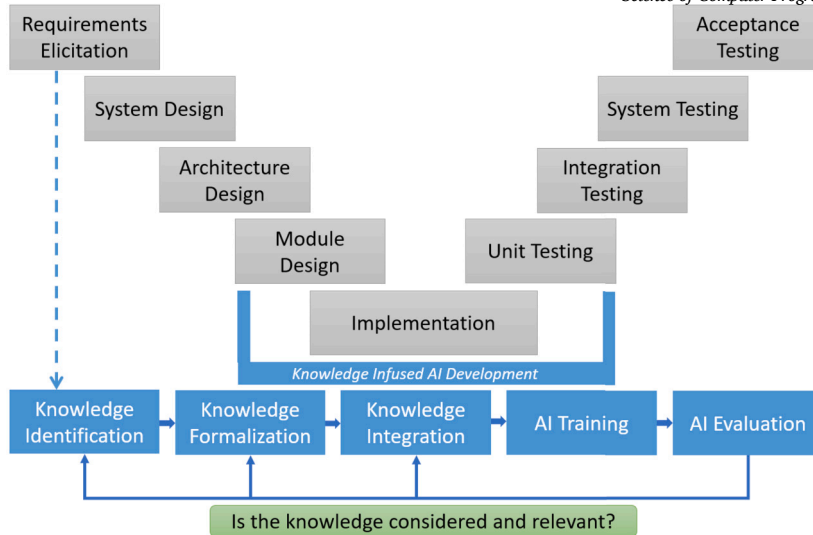


Fig. 2. Embedding of knowledge-infused AI development phases in the established V-model. Our contribution aims to determine whether identified and integrated knowledge is relevant in the sense that it enables an AI driving function to satisfy system requirements.

This paper is structured as follows: In Section 2, we summarize the motivation for knowledge-infused AI and introduce the phases of developing knowledge-infused AI models. In particular, we describe embedding the development into requirements-driven development of driving functions. Section 3 provides preliminaries for our work. In Section 4, we provide an overview of existing notions used in the Information Retrieval (IR) field and define a notion of relevance for knowledge infused into an AI. We present our approach for generating abstract knowledge scenarios based on Traffic Sequence Charts (TSCs) in Section 5. Section 6 presents our relevance testing procedure for knowledge-infused AI driving functions. A case study of this work is presented in Section 7. We discuss the related work in Section 8. Finally, in Section 9, we summarize our work and present future work.

2. Knowledge-infused AI driving functions

In 1994, Towell et al. presented one of the fundamental works on knowledge-infused AI¹ [26]. Their paper presents a hybrid learning system for neural networks. Connectionist learning techniques are combined with the infusion of problem-specific domain knowledge represented in propositional logic. It is demonstrated that the resulting networks generalize better than a wide variety of learning systems and even better than several techniques proposed by biologists [26].

Since then, technical innovations have enabled complex applications of knowledge infusion, e.g. in domains such as medicine [27] or transportation [28]. In the transportation domain, especially path planning, faces an utmost complex input space due to a rather unconstrained context in the real world. The following observation motivates the interest in knowledge infusion. Humans do not need to know all possible driving scenarios to behave appropriately in unknown situations. Instead, experiences, rules, and norms are used to derive decisions. In [10], the authors present a survey on approaches regarding infusing such relevant information, referred to as knowledge [10], into the AI. Such a knowledge infusion may improve the performance of cyber-physical systems that often face complex input spaces when making decisions, especially in non-data-covered scenarios.

2.1. Development of knowledge infused AI

In addition to a data-driven AI development process, the development of knowledge-infused AI comes with additional steps, each of which brings its challenges [10,18]. These steps are *Knowledge Identification*, *Knowledge Formalization*, *Knowledge Integration*, *AI Training* and *AI Evaluation*.

Fig. 2 gives an overview of these steps and illustrates their embedding in the system development process, which will be discussed later in Section 2.2. First, we briefly explain the individual steps.

Knowledge identification First of all, potentially relevant knowledge must be found. This phase is called *Knowledge Identification*. Here, domain experts and data scientists analyze the application domain for knowledge relevant to the system under development. In this phase, artifacts of analytical steps like the results of criticality analysis [29] may be used, which forms the basis for safety-critical knowledge. Identified knowledge may, for instance, be specific friction values for different road surfaces and weather conditions.

¹ often also referred to as knowledge-integrated, -augmented, or knowledge-based AI.

Knowledge formalization After identification, the knowledge is formalized for integration in the *Knowledge Formalization* phase. Many prior knowledge sources differ in quality, type of knowledge, and the form in which this knowledge is available. For instance, mathematical and physical knowledge is often already formalized and can be used for integration without much effort. In contrast, expert knowledge, e.g., traffic rules or court decisions, is usually not given inappropriately formalized for integration. To this end, researchers like Manas et al. [30], Szegedy [31], Westhofen et al. [33], and Borges et al. [34] explore how, e.g., naturally linguistic knowledge, can be formalized.

Knowledge integration and AI training Once knowledge has been formalized, it can be infused in different ways. Common are, e.g., the embedding in training methods like knowledge-infused Reinforcement Learning [13] or by utilizing the AI architecture [35]. This phase is called *Knowledge Integration*. Subsequently, after knowledge has been infused, the *AI Training* can take place. Depending on the integration and training methods, e.g., adjustments can be made to weighting knowledge at runtime to achieve better AI performance [18].

AI evaluation Finally, after training knowledge-infused AI, *AI Evaluation* then evaluates whether the trained AI driving function correctly applies infused knowledge and achieves the expected or a better performance. Different methods exist for evaluation. [32,36] present approaches to evaluation at runtime while [18] presents evaluation approaches performing executing the AI in tests. We also advocate assessing the relevance of the knowledge in this phase for three reasons. (1) Knowing what knowledge is relevant indicates whether the identification and formalization of the knowledge was correct. (2) We envision building knowledge bases to support the development of new AI models. Since training is expensive, such knowledge bases should only manage truly relevant knowledge. Finally, (3) we envision that methods for self-explanation can benefit from this knowledge.

2.2. Requirements-driven knowledge-infused AI system development

For the homologation of vehicles with knowledge-infused AI controlling a driving function, the AI development steps must be integrated into established requirements-driven development processes such as the established V-model development process, illustrated in Fig. 2. The phases of knowledge-infused AI development take place after the decision has been made that knowledge-infused AI shall realize a driving function. Hence, the phases can be seen as instances of the implementation phase of the V-model process. We envision that the *Knowledge Identification* uses results of the requirements elicitation since the AI incorporates knowledge relevant to the respective requirement. Whether the infused knowledge is relevant (enables the AI to satisfy its requirements) has to be checked in the phase of *AI Evaluation* – i.e., after knowledge formalization and the actual infusion.

3. Preliminaries

In this section, we give short introductions to the topics of artificial intelligence (Section 3.1), testing safety-critical systems (Section 3.2), and infusing knowledge into AI driving functions (Section 3.3). It also provides some background used in the later sections and introduces some basic notions and concepts.

In Section 3.1, some basic assumptions regarding the AI considered are made, and we introduce knowledge infusion as an operation that yields a new model from an initial model. Section 3.2 first explains the role of scenarios when testing safety critical systems and then explains the conceptual differences between formal verification and testing using simulation. In Section 3.3, we introduce our frequently used denotations, defining sets of trajectories and their relations.

3.1. Artificial Intelligence (AI)

This paper considers black-box AI methods. As learning approaches, we focus on the data-driven approaches of Supervised Learning and Reinforcement Learning, although the scope of the paper is wider than these. In *Supervised Learning*, each training example helps the model to recognize the differences between its prediction and the actual outcome. At the beginning of training, the model can make mistakes, but it improves its predictions through feedback from the labels and by adjusting the weights. Over time, the error decreases, and the model converges to a function that captures the relationship between inputs and outputs. Overfitting can occur when the model is too much tailored to the training data and generalizes poorly to new data. In *Reinforcement Learning*, the model receives feedback in the form of rewards, which can occur with a delay. The learning process is dynamic, as the agent actively explores the environment while continuously adapting. To achieve the best results, the agent must balance exploration (exploring new strategies or paths) and exploitation (taking advantage of knowledge already learned).

We classify as the reasons for AI failing to learn a requirement R as

Inadequate Model: The model lacks the capacity or the right architecture for the task.

Inadequate Data: The dataset is too small, poor in quality, or lacks some knowledge \mathbb{K} .

Inadequate Training: The training is either too short (underfitting), too long (overfitting), or there are inadequate settings of hyperparameters (e.g., bad gradient flow, poor choice of optimizer).

Knowledge infusion *Knowledge* means information validated by, e.g., experiments, studies, or experts [10]. In particular, knowledge does not need to be true in our setting.

As described in more detail in Section 2, we consider a basic *Knowledge Infusion Process* of the phases *Knowledge Identification*, *Formalization*, *Integration*, *AI Training* and *AI Evaluation*. In the following, we define the term *Knowledge Infusion Operation* as the operation that changes the initial model M_i to the infused model M_n . The Knowledge Infusion Operator hence represents the effect of *Knowledge Integration* plus *AI Training* and assumes identified and formalized knowledge as well as an initial model be given. If the infusion had no effect, the initial model M_i equals the knowledge-infused model M_n . The Knowledge Infusion Operation is parameterized by the integration means.

Notion (Knowledge Infusion Operation). The knowledge infusion operation, \odot , takes

- an initial net M_i ,
- knowledge K , and
- a means of integration \mathbb{I} , which is either
 - a modification of the architecture or
 - the knowledge as input
 as well as
- a training process P_t ,

and yields a new model $M_n = \odot (M_i, K, \mathbb{I}, P_t)$.

In abuse of notation, we also write $M_n = M_i \odot K$ to denote that M_n is the result of infusing knowledge K into M_i . \mathbb{I} and P_t are not further specified, but we make a further assumption: The training P_t of M_i causes a gradual shift of the model's focus as it refines its knowledge based on the new data distribution. As training continues, parts of the previously learned knowledge may be modified. The dynamics of how the previously learned is preserved, changed, or forgotten depends on several factors. One factor is the similarity of the data/requirement/knowledge. The model will likely preserve the original requirement if previously learned knowledge is similar or closely related to the new knowledge. If the new data is sufficiently different from the original training data, the M_n 's internal representation can shift significantly, which may lead to forgetting the previous knowledge. Another factor is the learning technique. To prevent catastrophic forgetting, continual learning techniques may be used, such as e.g., elastic weight consolidation (EWC), which adds a penalty for changing important weights tied to the old knowledge, or replay methods, which store samples from the previous task and periodically retrain the model on them to maintain older knowledge.

3.2. Testing

Part of the specification of a driving function is the *operational design domain (ODD)* that defines the limits for using the function. According to the ISO13586:2000 [22], scenarios are an intrinsic part of evaluating the safety of autonomous driving systems.

A *scenario* is a finite temporal sequence of an arbitrary number of situations. In contrast to a static traffic situation, scenarios enable the analysis of relationships and interactions between the environment, objects, and traffic participants and their evolution over time. Scenarios can be abstract to a certain degree and correspond to several concrete scenarios. In Section 5.1, we will present Traffic Sequence Charts (TSC), an example of a formal yet visual language that can be used to specify abstract traffic scenarios.

Due to the *open world context* of driving functions, we cannot explore all possible behaviors of an AI driving function. Therefore, a combination of testing and formal verification methods is employed to verify that the system satisfies a given requirement R . To test a system, it is the current state of practice that test engineers carefully determine a test suite for a given requirement R into a set of concrete test cases. This suite should cover all relevant cases wrt. satisfying R . Thus, it realizes a *good coverage of the ODD*. The *test executions (or test runs)* (i.e., executing the test cases) are monitored whether they satisfy R .

Test cases can be executed in simulation (i.e., within a virtual environment), in the real world, or in hybrid environments combining simulations with the real world. In this work, we focus on testing in simulation, but our contribution is not limited to this testing method.

3.2.1. Formal methods & testing in simulation

The notion *model-based design* refers to a design process where the development phases are accompanied by assessment methods that verify and validate the system under development early on. These methods either test the system in simulation or explore the system's behavior through formal methods.

The simulation engines may vary over time and have different foci as well as the employed formal models. We use in this paper the term *world model* to refer to the model \mathbb{W} of the application domain (either the formal model or model implicitly realized by the simulation engine). In our case, we assume that the world model describes the context of the driving function, which has a finite set of objects, such as vehicles or roads.

A *trajectory* $\tau \in \mathcal{T}$ of length l is a function assigning values to attributes of a finite set of objects (e.g., vehicles, roads) for each time $t \in [0, l) \subset \mathbb{T}$. $\tau(t)$ denotes a vector of values for all the objects' attributes at time $t \in \mathbb{T}$. We denote the set of all trajectories of the world model \mathbb{W} as \mathbb{W} . We also call the trajectories in \mathbb{W} *concrete traffic scenario*. In this paper, we study linear temporal properties, as described by, e.g., metric temporal logic [37].

Table 1
Commonly Used Denotations and their Meaning.

Denotation	Meaning
M_i	initial AI model
M_n	model after infusion of K
K	knowledge to infuse
K_i	knowledge learned by M_i
W	world model
R	requirement for M_i, M_n
\mathcal{S}	test suite
\mathbb{M}_n	set of traj. of $M_n \parallel W \subseteq W$
\mathbb{M}_i	set of traj. of $M_i \parallel W \subseteq W$
\mathbb{K}	set of traj. of W satisfying the knowledge $K, \mathbb{K} \subseteq W$
\mathbb{K}_i	set of traj. of M_i satisfying the requirement $R, \mathbb{K}_i \subseteq W$, represents the knowledge learnt by M_i
W	set of traj. of w
\mathbb{R}	set of traj. of W satisfying the requirement $R, \mathbb{R} \subseteq W$
\mathbb{S}	set of traj. of w that are part of $\mathcal{S}, \mathbb{S} \subseteq W$
$M \subseteq \mathbb{R}$	all of M 's traj. satisfy R
$M \subseteq_{\mathcal{S}} \mathbb{R}$	M satisfies R in \mathcal{S} , i.e. $M \cap \mathbb{S} \subseteq \mathbb{R}$
$M_n \geq_{\mathcal{S}, R} M_i$	M_n satisfies R in all test runs of \mathcal{S} where M_i satisfies R

We use the term *requirement* (or more generally *property*) to refer to a formal specification R that describes a set of trajectories of w . We denote the set of trajectories that satisfy R as \mathbb{R} . For example, R can be the requirement “Always respect safety distances” and is presented by all concrete scenarios where the safety distances are respected.

When we say our “AI model M satisfies the requirement R ”, it means that we place M into the world model w (see 3.3), denoted as w_M , and then all trajectories \mathbb{M} of w_M satisfy R , also denoted as $\mathbb{M} \subseteq \mathbb{R}$.

While formal methods are often able to derive that a property holds for all trajectories of w_M but often have to trade expressiveness for computational feasibility or even decidability, simulations are usually done non-exhaustively.

We can check whether a (simulation) test run satisfies a (formal) requirement R . Given a test suite \mathcal{S} , we say that “ M satisfies R verified by testing”, if we execute M in all runs of \mathcal{S} and they all satisfy R . We abbreviate verified by testing as vbt and also write “ $M \subseteq_{\mathcal{S}} \mathbb{R}$ ” instead of “ M satisfies R vbt”. Note that $M \subseteq_{\mathcal{S}} \mathbb{R}$ does not imply that all possible runs satisfy R , $M \subseteq \mathbb{R}$, since there may be runs that violate R but have not been chosen for the test suite \mathcal{S} .

3.3. Knowledge infusion & sets of trajectories

In the following, we introduce frequently used denotations. Table 1 gives an overview. This section can be skipped at the first read and used in the later sections.

When talking about knowledge infusion in this paper, we usually refer to the initial AI model as M_i . We denote the desired system requirement as R and the knowledge to be infused into M_i as K . Moreover, the AI model resulting from the knowledge infusion is usually denoted as M_n .

This paper discusses how knowledge infusion changes the behavior of the AI model semi-formally. Since we do not explicitly fix the world model (simulation engine), we cannot fully formally specify the resulting trajectories. Nevertheless, we can concisely express our ideas using sets of trajectories for reasoning about the accomplished system behavior. Hence, we present how these sets of trajectories could be defined in the following section. The sketched automata represent just one way in the AI model M , and the world model w could be specified.

Let automata M modeling the AI control and w modeling the world be given. The states of w are labeled propositions describing the values of the attributes of all objects of the world model. The edges of M and w are labeled by actions, and transitions are enabled based on whether respective guards are true. These guards refer to propositions describing the values of attributes of all objects of the world model w . When reasoning about the performance of an AI model M , we often refer to the set of trajectories that can occur when M is in control. Since we are interested in how well the AI model M performs its control task, we employ the controller realized by M to control the vehicle in the world model w , denoted as w_M . This means that we compose the automata modeling M and w , synchronizing w with M on the actions controlled by M . The resulting trajectories, denoted as \mathbb{M} , of the composed system $w_M = w \parallel M$ describe what can happen in the world w if M is in control. If $\mathbb{M} \subseteq \mathbb{R}$, then only behaviors that satisfy R can occur.

A test suite \mathcal{S} specifies a finite set of executions by fixing environmental attributes, i.e. the test conditions. We denote the set of trajectories of w that have the required environmental attributes as \mathbb{S} . $M_n \geq_{\mathcal{S}, R} M_i$ denotes that M_n satisfies R in at least all test cases of the test suite \mathcal{S} where M_i satisfies R .

Let us assume that M_i initially does not satisfy R , but there are some runs of M_i that satisfy R . Hence M_i has learned something but not sufficiently much. We use K_i to refer to the initial knowledge learned by M_i . More precisely, K_i is a constraint that specifies the set of trajectories \mathbb{K}_i of $M_i \parallel w$ that satisfies $\mathbb{K}_i \subseteq \mathbb{R}$.

4. Notion of relevant knowledge for AI driving functions

In this section, we explore the concept of relevant knowledge in knowledge-infused AI for requirement satisfaction in traffic scenarios and present related works. We define the notions of *ODD relevance* (see p. 8) and *Scenario-relevance Indication* (see p. 10), and discuss their relation and significance for the work presented here.

We define relevant knowledge as knowledge that enhances AI performance in meeting its requirements, R . More formally, relevant knowledge in AI-controlled driving functions refers to the knowledge essential for improving AI performance to fulfill the requirements R .

4.1. Why do we need a notion of relevance and a relevance test procedure?

Within the research project KI Wissen [7], academia and industry explored approaches on how different modalities of knowledge can be formalized and integrated into AI driving functions utilizing existing domain knowledge for data-driven AI driving functions. In three concrete use cases (i) pedestrian detection under occlusion, (ii) complex lane change, and (iii) controlled rule exception, the goal was to develop methods for integrating domain knowledge and validating the knowledge-infusions [18,14]. To this end, we investigated how domain knowledge can be formalized [14] and can check if M_n acts conform to $\mathbb{K}(M_n \subseteq_{\mathcal{S}} \mathbb{K}?)$ during runtime [32]. Thereby using Traffic Sequence Charts (TSC). Throughout this project, we discovered a lack of guidance in identifying relevant knowledge and choosing the means of knowledge infusion, whether this is the choice of training data or modification of the architecture. While this issue was not the focus of the project, and hence a state of practice has not been scientifically established within the project, Heyn et al. [19] recently investigated challenges encountered by practitioners when specifying training data and runtime monitors for safety-critical machine learning (ML) applications. They analyzed ten interviews with developers of ML models for critical applications in the automotive and telecommunications sectors, addressing two research questions:

“RQ1: What challenges do practitioners face when specifying training data for ML models in safety-critical software?” [19] and “RQ2: What challenges arise when specifying runtime monitors, particularly regarding the fulfillment of safety requirements?” [19].

Their findings include

- C1** that the data selection process is often nontransparent, with no clear guidelines for defining data variety or context, and current safety standards provide little guidance,
- C2** a lack of appropriate metrics and insufficient safety standard guidance hinders the specification of runtime monitors, and
- C3** challenges regarding explainability of ML systems [19, p.3].

Our concept of relevance addresses challenge **C1** by guiding the selection of data and design of AI driving functions, strengthening the connection between requirements and training data. Starting with an AI model M_i that does not satisfy its requirement R , we consider knowledge as relevant when its infusion enables the model to satisfy R . The process of infusing knowledge \mathbb{K} —transforming M_i into M_n —provides a causal explanation for why the infused model satisfies R . This approach establishes a clear link between requirements and training data during the knowledge infusion process.

Moreover, our notion of relevance accounts for various factors determining whether an infusion operation leads to satisfying R , offering guidance on when the knowledge can be reused. This serves as the foundation for a knowledge base that catalogs, curates, and maintains relevant knowledge, facilitating more efficient AI development through knowledge reuse. We discuss this issue in more detail on page 10.

Regarding the lack of guidance from safety standards in training data and runtime monitor specification (**C1** and **C2**), our approach uses the results from requirements elicitation, where a criticality analysis leads to safety-related requirements R_{safe} . Whether relevant knowledge is infused is tested by formalizing the infused knowledge \mathbb{K} and the associated safety requirements R_{safe} in abstract traffic scenarios, leading to formal and clearly defined conditions on the context and tasks of AI driving functions. This formalization also enhances transparency, allowing for cross-examination through formal methods or expert review. Thus, the infusion of relevant knowledge, as part of a requirements-driven development process, ensures alignment with safety standards and guides AI development in a structured manner. We can directly derive runtime monitors from the specified formal conditions, based on our work on Traffic Sequence Charts (TSC) runtime monitoring [32]. The degree of M_n satisfying R can be considered as a measure of M_n guaranteeing the respective safety properties. Our testing procedure hence provides a test-based measure of guaranteeing the safety properties.

Addressing **C3** on explainability, our relevance framework clarifies the causal relationship between the knowledge \mathbb{K} , and the models M_i and M_n , by explaining that M_i needs \mathbb{K} to meet R . While explainability is not the primary focus of this paper, we plan to explore self-explainable AI following the approach outlined in [21].

To summarize, the current development and training of ML models lacks guidance and transparency [19]. The infusion of relevant knowledge, as presented in this paper, contributes to alleviating this challenge by establishing a formally specified link between the requirements and data/knowledge infusion and characterizing influencing factors of relevance. It hence increases the reusability of knowledge and guides data selection for future ML models. The aforementioned advantages of our contribution are based on theoretical considerations. Any evaluation regarding building up a knowledge base would require long-term studies. In Section 7, several examples examine how our overall testing approach (cf. Fig. 1) establishes relevance indications. It thereby illustrates our clearly defined process of how different types of knowledge for safety requirements are formalized, infused, and evaluated regarding their relevance for the considered initial AI model.

4.2. Dimensions of relevance for AI driving functions

In the following, we discuss the term *relevance*. Note, that many notions of relevance have been discussed in the literature, and the discussion is ongoing. Hence, we briefly summarize the known dimensions that are important for our work and then add further new and specialized dimensions relevant to knowledge-infused AI. As relevance dimensions known from information retrieval (IR), we introduce *topic*, *system*, and *situation* for AI driving functions since these are important in our setting as well. We then introduce the dimension *predictability*, which originates from the field of AI. Finally, we define the new notion of *ODD relevance*.

The *multi-dimensionality* of relevance implies that “What knowledge is relevant to infuse into an AI controlling a driving function?” cannot be generally answered as such since it depends on multiple dimensions, such as the situation or the system.

Information retrieval (IR) deals with the retrieval of information from data storage systems, e.g., databases. Relevance has been widely conceptualized and discussed in IR from the 1960s to 1990s [38–41], but the discussion is ongoing – also because the IR systems are evolving.

In IR, a user has an information need specified as a user query. The information retrieved by the IR system should satisfy this need. The fundamental question of IR is hence *QIR*: = “What information is relevant to satisfy a user’s information need?”. Relevance is considered a relation between the retrieved information and the information needed but is also influenced by other aspects, such as the user’s cognition or the system’s processing capabilities. Relevance is hence called *multi-dimensional* [40].

In our work, we are interested in the question *QAI*: = “What information is relevant for an AI driving function in a given traffic scenario?”. In contrast to IR, an information need results from the requirements that the AI (or rather the system the AI is part of) must satisfy. While in IR the user is a human, the user is an AI in our setting. Analogously to IR, many aspects of the AI’s context influence what is relevant – such as the current environment, the state of the system, or the current behavior of other road users.

More precisely, we are here concerned with the question *QKI*: = “What knowledge needs to be infused into an AI so that it masters its driving function and satisfies the requirements?”. Our focus is on the knowledge that an AI internalizes during its training. We investigate the relevance of infused knowledge.² In our setting, the IR system becomes the knowledge base, which is the result of the previous development phases (*Knowledge Identification* and *Formalization*). While a user formulates a query in IR, the quest for information is done in *Knowledge Identification* by domain experts. In the context of AI driving functions, the need for information/knowledge results from the goal of satisfying the requirements. The dimensions of relevance most important for the work presented in this paper are summarized in the following.

Topicality. In IR, topicality is a relation, the topic match, between a topic of a query and a topic of a retrieved document (cf. [41,42]). Transferred to our setting, it is the relation between the topic of a requirement and the topic of knowledge. For example, topical relevant for satisfying the requirement “Keep a distance between 2-15 meters to a static obstacle” is knowledge about the vehicle’s physical dynamics, in particular the effects of deceleration.

System Relevance. System relevance is a relation between a requirement, the knowledge, and the system (that is, a specific AI model in a vehicle). This notion is inspired by the notion of system relevance from IR, which refers to the relation between a query, the retrieved document, and the internal organization of a system (cf. [43,42]). The AI’s architecture, number of neurons, activation functions, etc., and also the training data set, duration, and training method influence a trained AI’s performance. The notion of system relevance emphasizes that these factors influence what is relevant. For instance, after training, we have a different system, and for this system, new knowledge becomes relevant, given that it has internalized the initial knowledge.

Situational relevance. In IR, situational relevance emphasizes that the current situation in which the user is in (cf. [44,40]) influences what is relevant. The influence of the situation on the relevance is certainly high, considering driving functions. To reflect this influence more accurately, we define a specialized notion of situational awareness below.

Predictability. Predictive relevance refers to the importance of a data point or pattern contributing to accurate predictions when training an AI model [45]. In this work, we are interested in a related notion. We instead are targeting formally specified knowledge that can drive the generation of data sets.

In this paper, we are concerned with AI models controlling driving functions. These driving functions are developed for a certain operational design domain (ODD). If the domain is exited at runtime, a different function (or the user) takes over. The ODD limits the situational dimension and thus influences what knowledge is relevant.

When we want to characterize what knowledge is relevant for the infusion operation, we have to consider that the relevance of knowledge is always influenced by the initial model M_i into which knowledge is going to be infused, the integration means \mathbb{I} , and the training process P_t (see Section 3.1).

Notion (ODD-relevance). ODD-relevance is a relation between a requirement R , a model M_i , and knowledge K . We say that the *knowledge infusion operation* \odot is *ODD-relevant* for M_i , if

- M_i does not satisfy R in the ODD.
- We can infuse K into M_i (i.e. find \mathbb{I} and P_t and apply \odot), so that
- the resulting trained model $M_n = \odot(M_i, K, \mathbb{I}, P_t)$ satisfies R in the ODD.

² Knowledge means information validated by, e.g., experiments, studies, or experts [10].

For example, knowledge \mathcal{K} of how to calculate safety distances when it rains will usually be ODD-relevant for a road vehicle, the requirement $R = \text{“Keep a distance between 2-15 meters to a static obstacle”}$ and to a model M_i still lacking specific knowledge about how to compute the safety distance when it is raining. Note that the above notion requires that we can infuse knowledge, which means the notion requires the existence of $\mathcal{I}, \mathcal{P}_i$ such that the trained model satisfies R .

The above notion calls a *sufficient* knowledge infusion relevant. It does not require minimal knowledge; that is, we might be able to infuse a less informative \mathcal{K}' into M_i , resulting in model M_n' that satisfies R as well.

Similarly, a knowledge infusion might not be ODD-relevant but *partly ODD-relevant*. We call a knowledge infusion partly ODD-relevant if the retrained model M_n does not satisfy the requirement R but is strictly more successful than M_i wrt. R . More formally, let R_i be the requirement satisfied by M_i with $R \Rightarrow R_i$. If M_n satisfies R' and $R \Rightarrow R'$ and $R' \Rightarrow R_i$, we call the knowledge-infusion *partly* ODD-relevant.

4.3. Assessment of relevance: testing for relevance indications

Along with discussions of “What is relevance?” the question of “How to measure relevance?” has been discussed. Schamber et al. stated “[...] Relevance is a complex but systematic and measurable concept if approached conceptually and operationally[...]” [41] in 1990 referring to relevance in information retrieval. The multiple dimensions of relevance, as mentioned in the previous section (the internal realization of the system, its current state, the user’s cognitive abilities, the situation etc.), make it difficult to measure relevance precisely because some dimensions are not directly observable.

In Section 4.3.1, we discuss the challenges of measuring the relevance of knowledge in our setting. In our opinion, the key challenges are

- (i) ODD-relevance refers to the performance of the AI system in the real world,
- (ii) whether knowledge is relevant also depends on the knowledge infusion operation and, hence, on M_i , \mathcal{I} and \mathcal{P}_i , and in particular
- (iii) the effect of knowledge infusion is doxastic (see below) and hard to assess since, more often than not, it changes a black box system.

In Section 4.3.2, we define the notion of *Scenario-relevance Indication*. This notion derives relevance indications from the runs of an AI that are observed when executing a test suite of a given scenario. The notion links efficiently computable (i) relevance indications and (ii) ODD relevance via the established approach of executing scenario-based test suites.

4.3.1. Assessing the doxastic effect of knowledge infusion

In order to explain the difficulty of measuring the relevance of infused knowledge, we describe the doxastic (referring to beliefs in the sense of doxastic logic) effect of knowledge infusion. Therefore, we distinguish between the *real world*, i.e., the application domain, and the *internal world model* of the AI system, in terms of which the AI expresses its beliefs.

In a nutshell, an AI system makes decisions based on its *beliefs*, which are built based on *observations* and its *knowledge* about the world. Knowledge infusion causes an AI model to change its beliefs or even the internal world model.

Fig. 3 illustrates the effect of knowledge infusion on the AI’s beliefs that result in observable real-world behavior. The Venn diagrams show set relations of real-world behavior at the top and “believed” behavior at the bottom. Let us assume we have a model M_i that does not (entirely) satisfy its requirements R , denoted as $M_i \not\subseteq R$.

The top of (a) shows M_i in terms of its behaviors within its environment. Some of M_i ’s behaviors satisfy R , but not all. During its initial training, M_i has deferred from the training data some knowledge \mathcal{K}_i about the environment and the requirements it must satisfy. The bottom of (a) depicts M_i ’s beliefs about its environment and requirements. These do not perfectly match with the reality (top). We assume here that M_i “believes” to satisfy the requirement (bottom), but it does not satisfy the requirements in the real world (top).

(b) illustrates the infusion of relevant knowledge that causes M_n to build new beliefs and thus causes M_n to satisfy the requirement R , $M_n \subseteq R$. At the bottom, the new beliefs about the environment and requirements are depicted, which now match better with reality (at the top). Note that M_n “believes” in (a) and (b) to satisfy the requirement (bottom), but since only in (b) belief and reality match better, it actually satisfies the requirement in the real world (top).

The bottom line is that knowledge changes an AI’s internal beliefs, and consequently, it behaves differently in the real world, reacting to its perceived environment. These changes are complex to assess,³ but they determine whether a model can generalize or is overfitted. For black box models, we can only monitor the AI’s observable performance to infer whether knowledge \mathcal{K} has been infused (see Section 6, Knowledge Infusion Test), whether M_n loses valuable knowledge of the initial model M_i (see Section 6, Knowledge Preservation Test) and whether \mathcal{K} causes the AI to satisfy the requirement R (see Section 6, Requirement Satisfaction Test).

However, as discussed in Section 3.2, the whole system behavior cannot be explored. Instead, simulations of test scenarios (i.e., scenarios of a test suite) that cover the ODD are executed and monitored. Hence, we derive indicators that indicate relevance but cannot guarantee it. However, given good coverage of the ODD and a well-chosen test suite for each scenario, the relevance indicators are an efficient way to derive relevance within an established development process.

³ there is ongoing research to self-explainability of AI.

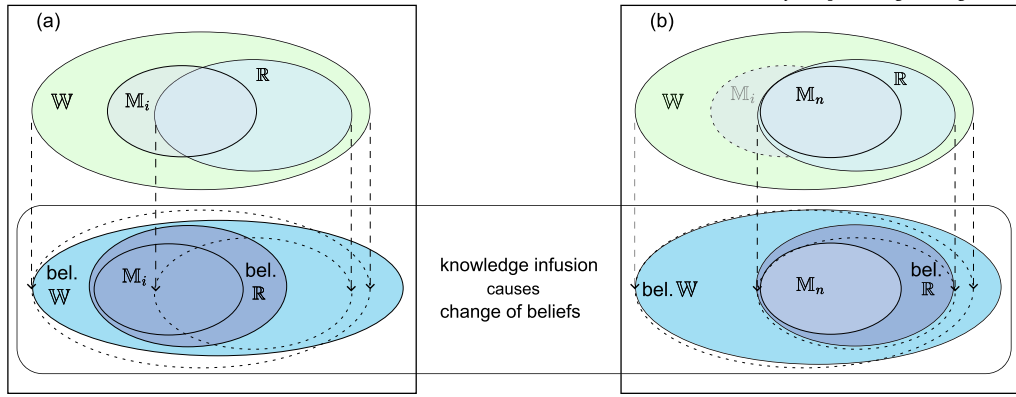


Fig. 3. Effect of knowledge infusion. Knowledge infusion changes the beliefs the AI builds (bottom) and, in turn, changes its behavior in the real world (top).

Above, we explained the doxastic effect in terms of the conceptual framework for the relevance of safety-critical autonomous systems as used in [47]. While Rakow in [47] is mainly concerned with the question “What knowledge and observations of the world are relevant for a given system (with a certain belief space) to accomplish its mission successfully?” focusing on the design of information retrieval capabilities, we are here interested in “What knowledge is ODD-relevant to infuse into an AI model so that it satisfies the requirement R of the driving function?” and hence focusing on the training phase and its evaluation.

4.3.2. Scenario-relevance indication

Since we cannot explore all behaviors of an AI model within its ODD, we derive indications from running test suites of scenarios in our simulations. As discussed in Section 3.2, the scenarios and their test suites for a given requirement R are specified by a test engineer.

Recall that we say M_i satisfies R vbt in $Scen$, if M_i satisfies R in all runs of the test suite of the scenario $Scen$ (cf. p. 6). Otherwise, we say M_i does not satisfy R vbt in $Scen$.

Notion (Scenario-relevance Indication). Let a requirement R , a model M_i , a scenario $Scen$, and a test suite \mathcal{S} be given.

We say that we have an *indication that the knowledge K is relevant* for M_i to satisfy R in the scenario $Scen$, if

- M_i does not satisfy R in $Scen$ vbt,
- we can apply a knowledge infusion to M_i (i.e. we find I and P_i and apply \odot), such that
- the resulting model $M_n = \odot(M_i, K, I, P_i)$ satisfies R in the scenario $Scen$ vbt.

This notion bridges the real world and scenario-based testing. We have found a scenario-relevance indication when we can verify by testing that the knowledge infusion of K makes M_i satisfy the requirement R . Next, we discuss how to derive ODD-relevance indications from scenario-relevance indications.

Scenario relevance indications and knowledge bases for ODDs Knowing whether a given piece of knowledge is scenario-relevant is already valuable. However, to build up a knowledge base, our ultimate goal is to determine whether certain knowledge K is ODD-relevant. A scenario-relevance indication of K can, as such, be considered an *indication* of ODD-relevance as well, given that the scenario is part of the ODD. How *strong* this indication is certainly depends on the test suite and the fit between simulation and real environment. Both aspects are common hurdles that are faced as part of the design process. We envision that indications can be collected and that, over time, more precise specifications of relevant knowledge will hence become available, and stronger indications can be derived by agglomeration.

The notions of ODD-relevance (cf. p. 8) and scenario-relevance (cf. p. 10) both assume that an infusion operation can be applied to infuse K i.e., $\odot(M_i, K, I, P_i)$ where M_i is the initial model, I the means of integration and P_i the training process. A database of relevant knowledge K could catalog tuples $(M_i, K, I, P_i, R, \mathcal{S})$, i.e., the infusion operation, the requirement, and the test suite. When using the database, the similarity between AI models, knowledge infusion operation, and requirement has to be judged, and the strength of indication can be derived from the test suite \mathcal{S} . It is out of the scope of this paper to discuss all these issues. However, the user should be aware that although relevance indications will lead to a more focused knowledge search and retrieval, the knowledge base must be used, considering these aspects.

Given a requirement, a systematic approach to determining an initial AI model and a systematic procedure for the knowledge infusion operation will strengthen the meaning of relevant indications for the newly developed AI. To this end, we present a systematic approach for knowledge infusion in Section 5.

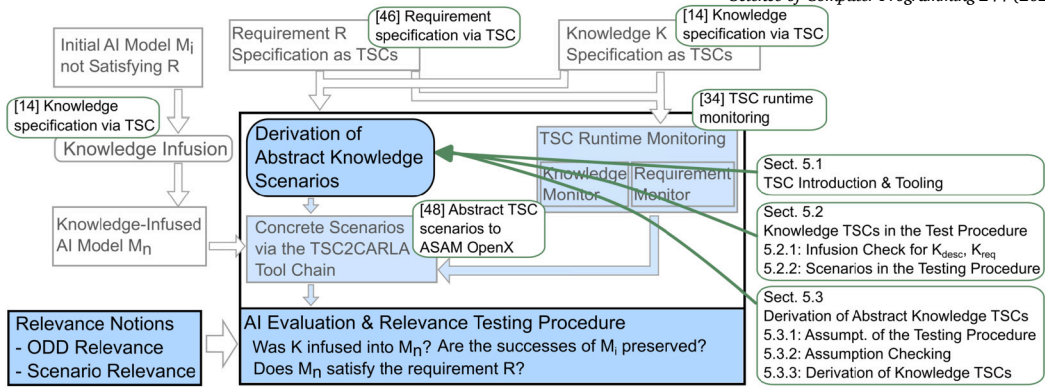


Fig. 4. Overview of our contribution as in Fig. 1 with the table of contents of Section 3.2 and the employed TSC tool & research landscape.

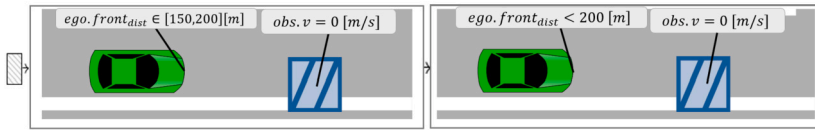


Fig. 5. A TSC of three snapshots. The first snapshot is a True Chart, which expresses that anything may happen for an arbitrary non-zero amount of time. The second snapshot expresses that the ego car faces an obstacle (object with velocity zero) with a distance of 150 to 200 m. Then the ego car has a distance of at most 200 m.

5. Systematic derivation of abstract knowledge scenarios

As illustrated in Fig. 4, our testing procedure to determine whether infused knowledge is relevant uses abstract knowledge scenarios. These knowledge scenarios combine requirements and knowledge. In this section, we explain how we construct the abstract knowledge scenarios and how they are used within our testing procedure to guide the data selection and training and to derive runtime monitors. The key to these benefits is using the formal specification language. We have chosen Traffic Sequence Charts [24], which provide a visual, formal, and intuitive specification language that also allows the employment of a rich tool landscape. We next give a brief introduction to TSCs

5.1. Traffic sequence charts

TSCs are a formalism to specify spatio-temporal logic properties in terms of sequences of constraints. TSCs use a time model \mathbb{T} , which allows the specification of traffic scenarios with continuous and discrete time semantics. Each constraint holds for a non-empty time duration, and consecutive constraints hold contiguously. Their most used visualization is called *Spatial View (SV)*. A spatial view snapshot formalizes a conjunction of propositional constraints and focuses on an intuitive visualization of the spatial aspects. A TSC (specification) basically is a sequence of SVs. A simple example is given in Fig. 5. Below, we explain this more precisely.

Basic and composed charts In order to specify constraints on a trajectory⁴ in different phases of a traffic scenario, there is the concept of *Charts*.

The simplest one is the so-called *Basic Chart*, which contains an *Invariant Node*. In our context, the Invariant Node shows a Spatial View (SV) so that it specifies propositional constraints that invariantly hold over a period of time, i.e., for the interval $[b, e] \subseteq \mathbb{T}$. In Fig. 5, the invariant of the second snapshot specifies “ego car is facing an obstacle and has a distance of 150 – 200m”. A Basic Chart thereby describes properties of *trajectories* (or *concrete traffic scenarios*) over the underlying world model w (cf. Section 3). A *True Chart* is a special form of Basic Charts. It specifies that any behavior is allowed within a non-empty interval. It is visualized as a grey hatched rectangle (cf. the first snapshot of Fig. 5).

To specify more complex abstract traffic scenarios, Basic Charts are composed of more complex structures using operators (Sequence, Concurrency, Choice, etc.). We call these *Composed Charts*. Fig. 5 shows a Composed Chart of three Basic Charts.

Premise A *Premise* can be combined with a Composed Chart to express, e.g., an implication. Implications are particularly suitable for the specification of requirements and scenario-relevant knowledge in combination with scenario-based development. A *Premise* can have just a *History* or a *History* and *Future*, which can be a Composed Chart. We call the former *History-implies-Consequence* (HiC) and the latter *History-and-Future-implies-Consequence* (HaFiC). A first example of a HiC is given in Fig. 6 on page 14. The semantics and visual syntax of HiC and HaFiC are summarized in Table 3. For more details, we recommend the introductory paper [24].

⁴ Recall that a trajectory is an assignment of values to all attributes of a finite set of objects at each point in time $t \in [0, l]$ (cf. Section 3).

Table 2
Notation, semantics and visual syntax of *Basic Charts*.


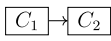
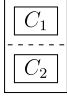



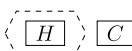
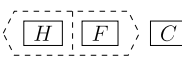
Name	Notation ($C =$)	Semantics (C satisfied on τ from time b to e iff)	Visual Syntax
Invariant Node of a Spatial View sv	$[sv]$	$e > b$ and sv satisfied by $\tau(t)$ for all $t \in [b, e]$	
Sequence of Basic Charts C_1 and C_2	$(C_1; C_2)$	$\exists m \in [b, e]$: C_1 satisfied by all $\tau(t)$, $t \in [b, m]$, and C_2 satisfied by all $\tau(t)$, $t \in [m, e]$	
Concurrency of Basic Charts C_1 and C_2	$(C_1 \& C_2)$	C_1 satisfied by τ within $[b, e]$ and C_2 satisfied by τ between on $[b, e]$	
Choice of Basic Charts C_1 and C_2	$(C_1 C_2)$	C_1 satisfied by τ between $[b, e]$ or C_2 satisfied by τ between $[b, e]$	
Negation of Basic Chart C_1	$!C_1$	between $[b, e]$ C_1 is not satisfied by τ	
Basic Chart C_1 with Duration Constraint $\bowtie d$	$\bowtie d(C_1)$	$e - b \bowtie d$, $\bowtie \in \{<, >, =\}$ and C_1 satisfied by τ on $[b, e]$	

Table 3
Syntax and semantics of a TSC with activation mode always.

Name	Semantics (satisfied by a trajectory τ of length l iff)	Visual Syntax
HiC-TSC with Basic Charts history H and consequence C	$\forall 0 \leq b \leq m \leq l : H$ satisfied on τ between b and $m \Rightarrow \exists e \geq m$ of the τ : C satisfied on $[m, e]$	
HaFiC-TSC with Basic Charts history H , future F and consequence C	$0 \leq b \leq m \leq e \leq l : H$ satisfied on τ between b, m and F is satisfied on τ between $m, e \Rightarrow C$ satisfied on τ between m, e	

Activation mode In this paper, we use a slightly simplified version of TSCs and omit the so-called bulletin board.⁵ We thus mention the activation mode of a TSC separately here. The activation mode specifies when the chart constraints of the TSC must hold on a trajectory τ .⁶ Any chart constraint C must be satisfied at all times t between a begin time point b to an end time point e , i.e. $\tau(t)$ must satisfy C for all $t \in [b, e]$. If the activation mode is *initial*, the TSC has to hold initially; that is, the chart constraints must hold from $b = 0$ and up to a time point e greater b . If the activation mode is *always*, the chart constraints must hold along τ at all $b \in \mathbb{T}$, and for each b , there has to be an end time e greater b .

5.1.1. TSC scenarios & TSC tooling

A TSC specifies an *abstract scenario*, i.e. it represents an arbitrary number of concrete scenarios (i.e. trajectories of the underlying world model).

As illustrated in Fig. 4, we use our test procedure and TSC tooling to derive concrete scenarios. These scenarios can be used for training during knowledge infusion and for testing, which is our focus. The approach of Becker et al. [48] allows to derive concrete scenarios from abstract TSCs specifications. A more detailed description of the implementation with a focus on generating reasonable test suites can be found in [46]. Thereby, it is possible derive concrete scenarios in the form of ASAM OpenX [49] files, which are directly simulatable, e.g., in simulators such as CARLA [50].

Also illustrated in Fig. 4, we employ runtime monitors that observe whether the requirement R is satisfied and whether the knowledge is infused. Grundt et al. presents in [32] how these monitors can be derived from a TSC. A more detailed description of the implementation focusing on the runtime monitoring of complex system requirements is submitted in the same special issue as this work.

⁵ It declares, for instance, objects and the activation mode of a TSC.

⁶ i.e. a concrete run of the W (or simulation engine, respectively), cf. Section 3.3.

Table 4
Overview of Introduced Denotations.

Name	Role	Description
R	requirement	implication of the form “ $R_{desc} \Rightarrow R_{cons}$ ”; intuition: in the cases R_{desc} , R_{cons} required; given as HiC- or HaFiC-TSCs
R_{desc}	R’s premise	premise of the requirement R, given as TSC, describes the cases where R_{cons} is required consequence of requirement R, given as TSC, describes what is required (in the given circumstances R_{desc})
R_{cons}	R’s consequence	
M_i	initial AI model	initial AI model trained to satisfy R
M_n	retained AI model	M_n evolves from M_i by retraining with the goal of infusing knowledge \mathcal{K}
\mathcal{K}	knowledge to infuse	M_n is M_i being retrained with \mathcal{K} to satisfy R; we assume that \mathcal{K} has the form of \mathcal{K}_{req}
\mathcal{K}_{req}	knowledge req.	implication of the form “ $\mathcal{K}_{desc} \Rightarrow \mathcal{K}_{cons}$ ”; intuition: in the cases \mathcal{K}_{desc} , \mathcal{K}_{cons} required; given as HiC- or HaFiC-TSCs
\mathcal{K}_{desc}	\mathcal{K}_{req} ’s premise	premise of the knowledge \mathcal{K}_{req} ; describes the cases that have been identified as critical consequence of the knowledge \mathcal{K}_{req} ; describes the required constraints that have to be accomplished in the critical cases \mathcal{K}_{desc}
\mathcal{K}_{cons}	\mathcal{K}_{req} ’s consequence	

Their intuitive visual specification is a major benefit of TSCs for *Knowledge Identification* and *Formalization*. In [25], TSCs have hence already been used to specify and formalize system requirements as well as for the purpose of knowledge specification for knowledge infusion of AI [14]. An intuitive visualization fosters communication among experts of different disciplines and hence allows for cross-checks. Considering the responsibility shifts from humans to AI, further and new experts will be involved in the system development. These experts are likely not to comprehend temporal logical formulae easily. Thus, the visualization enables the experts to diagnose incorrect knowledge specifications.

5.2. Abstract knowledge scenarios in the test procedure

This section gives an overview of the testing procedure based on a running example. It prepares the more detailed presentation in Section 5.3, where we describe the systematic derivation of abstract knowledge TSCs.

In Section 5.2.1, we illustrate that we infuse *environmental descriptive* and *requirement knowledge*. Both specification styles will lead to knowledge TSCs \mathcal{K}_{req} and \mathcal{K}_{desc} that will be used in our testing procedure. We discuss these knowledge specification styles based on a running example. This example will be treated more formally in the following sections. In Section 5.2.2, we sketch how the testing procedure uses the TSCs \mathcal{K}_{desc} , \mathcal{K}_{req} , R, and TSC tooling to establish whether a given knowledge is relevant.

5.2.1. Knowledge scenarios and knowledge infusion

When an AI model M_i fails to satisfy its requirements R, the infusion aims to make the retrained AI model M_n satisfy R. For the following, we assume that knowledge \mathcal{K} has been identified as relevant and has been formalized in the previous phases *Knowledge Identification* and *Knowledge Formalization*. In the following, we consider the case that retraining of M_i will use training data that represents \mathcal{K} .

For the following, we assume, moreover, that R is given as HiC- or HaFiC-TSCs. For simplicity and without loss of generalization, assume that \mathcal{K} has the form of \mathcal{K}_{req} and is also a HiC or HaFiC-TSC. We refer to R’s *Premise* as R_{desc} and to its *Consequence* as R_{cons} . Likewise, we refer to \mathcal{K} ’s *Premise* as \mathcal{K}_{desc} and to its *Consequence* as \mathcal{K}_{cons} . Table 4 gives an overview of the used denotations.

Example. (Requirement, Knowledge & Model) As a running example, let us consider the requirement $R = \text{“Always stop in a distance of 2-15 meters to a static obstacle”}$. Let us assume that the initially trained model M_i that does not satisfy R. It especially violates R in scenarios where the friction is decreased due to rain. Further, we assume that inspection of the training data shows that these scenarios were under-represented.

For our approach, the knowledge can simply state $\mathcal{K}_{env} = \text{“At some times it is rainy”}$. It could also specify required behavior in terms of $\mathcal{K}_{req} = \text{“If it is rainy, brake taking the reduced friction into account”}$. We choose to consider \mathcal{K}_{req} here. In order to infuse \mathcal{K}_{req} into M_i , we enrich the training data set by synthesized concrete scenarios where it is rainy. In the case of Supervised Learning, we label concrete scenarios where \mathcal{K}_{req} holds as positive and scenarios where \mathcal{K}_{req} is violated as negative. In the case of Reinforcement Learning, we choose an appropriate reward function [14] that rewards sufficiently early breaking on rainy roads. \square

We differentiate between specifying descriptive knowledge, \mathcal{K}_{env} , about the environment (*env-desc*) and specifying knowledge regarding the required behavior, (*req*). Both specification styles (*env-desc*) and (*req*) work for our approach and we can unify them by deriving a knowledge specification \mathcal{K}_{desc} that characterizes the scenario context and knowledge specification on a (refined) requirement \mathcal{K}_{req} .

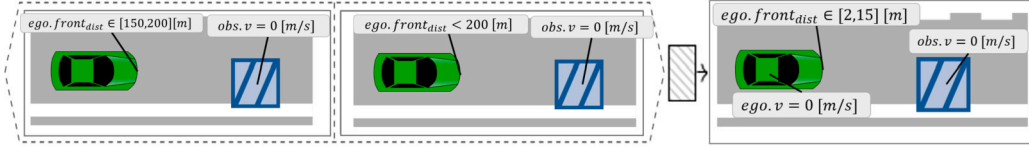


Fig. 6. Requirement: “Always stop at a distance between 2-15 meters to a static obstacle.”.

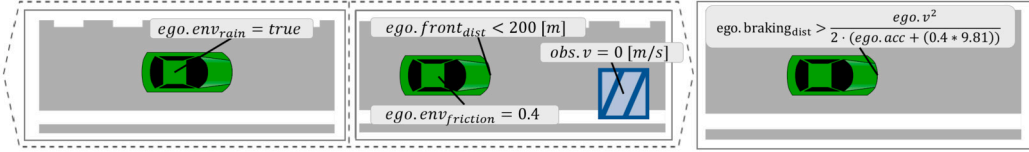


Fig. 7. Knowledge K_{req} : “Always if it is rainy and the friction is then reduced, brake taking the reduced friction into account.”

In our running example, the unification leads to the same two knowledge scenarios for both knowledge styles, (*env-desc*) and (*req*). We derive that we have to infuse the knowledge $K_{desc} = \text{“There are scenarios where it is rainy”}$ and scenarios that illustrate $K_{req} = \text{“If it is rainy, then brake and take the friction into account”}$.⁷

5.2.2. Testing for relevant knowledge

In order to answer “Is K relevant for infusion M_i , so that the new model M_n then satisfies requirement R ?” our testing procedure in Section 6 does three tests Knowledge Infusion Test, Knowledge Preservation Test and Requirement Satisfaction Test. In this testing procedure, abstract TSC scenarios play a central role.

- The Knowledge Infusion Test checks whether knowledge K has been infused. To this end, we check whether M_n satisfies the knowledge K_{req} within the identified scenario context K_{desc} . In other words, we check whether M_n has grasped the refined requirement on the infusion training data compilation.
- The Knowledge Preservation Test checks whether M_n loses valuable knowledge of the initial model M_i . To this end, the tests previously run on M_i are rerun to check whether M_n is a real improvement compared to M_i . In this test, the monitors for the initial requirement R are used to evaluate M_n ’s performance.
- The Requirement Satisfaction Test checks whether K causes the AI to satisfy the requirement R specified as TSC.

Example. The requirement R of our running example can be specified via a HaFiC as illustrated in Fig. 6. The activation mode is *Always*. The first SV snapshot specifies the History of the *Premise*. It shows that the controlled car *ego* is in front of a static obstacle in its lane. The second snapshot of the *Premise* specifies the future. It will be at a distance of less than 200 m, i.e., still in front of the obstacle on the same lane. The third and the fourth snapshot build the *Consequence*. It expresses that (if *ego* starts facing the obstacle and will stay there, then) *ego* must eventually come to a stand-still at a distance of 2-15 meters from the obstacle. The third snapshot is a True Chart encoding “eventually”.

The knowledge K_{req} can be specified via HaFiC-TSCs as visualized in Fig. 7. The activation mode is *Always*. The first SV snapshot in the *Premise* expresses that it is raining within *ego*’s environment. While the second SV snapshot in the *Premise* expresses that *ego*’s friction is reduced and a static obstacle is close by. The *Premise* hence expresses that first, it is raining, and then *ego* approaches a static obstacle while the friction is reduced. The *Consequence* snapshot expresses that then *ego* has to start braking, taking into account the reduced friction. \square

5.3. Derivation of abstract knowledge scenarios

In Section 5.3.3, we describe the general process of using TSCs to specify abstract knowledge scenarios for knowledge infusion and relevance testing. But before that, we list assumptions that we make for the relevance testing procedure (Section 5.3.1) and explain how we can check whether these assumptions hold (Section 5.3.2).

5.3.1. Assumptions for the relevance testing procedure

For our testing procedure, we assume that the AI model is developed in a process where knowledge identification and knowledge formalization have already been made. We assume to have an initial model M_i and a retrained model M_n , requirements R , infused knowledge K , and a scenario S_{scen} for which the models are trained. We assume that R , K , S_{scen} are specified as TSCs. We use the denotations as in Table 2, p. 12, and Table 4, p. 13. We moreover make the following assumptions:

⁷ For (*env-desc*), K_{req} is derived by combining K_{desc} and R .

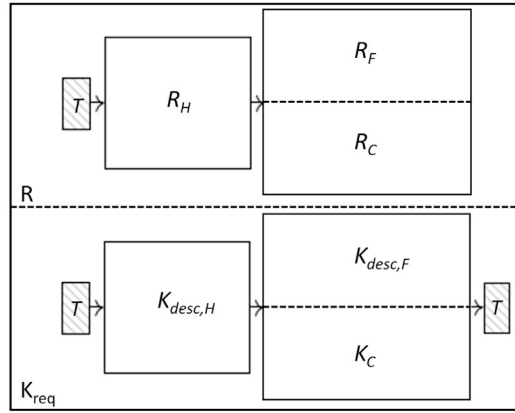


Fig. 8. Abstract Knowledge TSC. Composition of the additional requirements K_{req} and requirement R to describe the refined requirement for M_n .

(Impl) Given requirements are formalizable as HiC or HaFiC TSCs.

We assume that requirements are formalized as TSCs in form of implications; that is, the resulting TSC is a HiC- or HaFiC-TSC with activation mode *always*.⁸

(Scen) The knowledge is formalizable and considers a subscenario of $Scen$.

In knowledge identification and formalization, knowledge that is potentially scenario-relevant is identified that is supposed to lead M_i to satisfy R . Therefore, knowledge K_{desc} identifies additional constraints on the scenario $Scen$.

(Ego) The *Consequences* of both requirement specifications R and K_{req} specify the expected behavior of the ego vehicle.

(Ref) K_{req} does not contradict R .

We assume the additional requirement K_{req} does not contradict the initial requirement R . Since K_{req} should lead to M_n to satisfying R , their conjunction, R and K_{req} , should refine R .

Note that based on these assumptions, we assess the relevance of infused knowledge for a given AI model.

5.3.2. Abstract knowledge scenarios for testing requirement refinement

In this section, we describe how to check whether (Scen) and (Ref) holds. Since errors in the formalization of κ can happen, these checks can provide feedback to the *Knowledge Formalisation Phase*.

To ensure “The knowledge K_{desc} considers a subscenario of $Scen$ ” (Scen), we examine we can synthesize concrete scenarios that satisfy knowledge scenario K_{desc} as well as the scenario $Scen$.⁹ Therefore we use a world model \mathbb{W} ¹⁰ that generates only concrete scenarios of $Scen$. As the simulation engine can be seen as the world model in our context, it basically means that the simulator realizes a world as described by the abstract scenario $Scen$. We then check whether we can synthesize a concrete scenario satisfying K_{desc} . If K_{desc} is inconsistent with the scenario $Scen$, there is no concrete scenario $Scen_{K_{desc}}$ of world model \mathbb{W} (or the simulation engine) satisfying K_{desc} . We hence can use the synthesis approach of Becker et al. [48]. In our running example, we have the knowledge $K_{desc} = \text{“Sometimes it is rainy.”}$ If we mistakenly use a world model where it cannot rain, no concrete scenario is synthesizable for K_{desc} .

To show “ K_{req} does not contradict R ” (Req), we construct a TSC that composes K_{req} and R , as described below and illustrated in Fig. 8. We then check whether the constructed TSC contains logical inconsistencies or physical implausibilities and check whether a concrete scenario can be synthesized [48].

Fig. 8 illustrates how the requirement R and knowledge K_{req} , i.e. the additions to the requirement R , are composed. The top part thus abstractly specifies scenarios that satisfy the requirement R and satisfy R ’s precondition. The bottom part analogously abstractly specifies scenarios that satisfy the knowledge requirement K_{req} and satisfy K_{req} ’s *Premise*. Recall that K_{req} ’s premise is K_{desc} comprising History and Future.

Since no triggering precondition is needed, the TSC of Fig. 8 does not have a *Premise* but is the *Concurrency* of two *Composed Charts*, R and K_{req} . Both, R and K_{req} of Fig. 8, start with a *True Charts*. At the top, the History Composed Chart of R (R_H) is followed by the Concurrency of the Future of R (R_F) and its Consequence (R_C). The True Charts encode that we do not assume K_{req} to be synchronized with R , but that it adds to R in some way.

⁸ Note that (Impl) assumes that the activation mode is *always*. We make this assumption to simplify the following presentation. It is also possible to consider the activation mode initial. The TSCs can be derived analogously.

⁹ If K_{desc} allows scenarios outside of $Scen$, we use $K_{desc}' := Scen \ \& \ K_{desc}$.

¹⁰ i.e., the world model underlying the TSCs cf. Section 5.1.

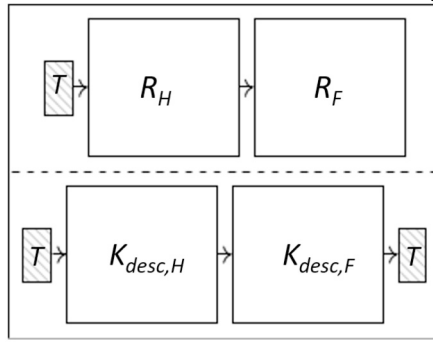


Fig. 9. Abstract Knowledge TSC. Composition of the knowledge requirement K_{req} and requirement R to describe the refined requirement for M_n , within the identified relevant subscenario.

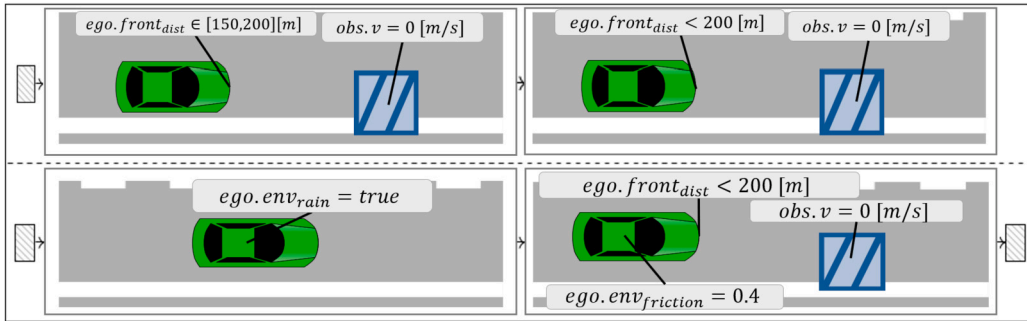


Fig. 10. Derivation of abstract knowledge scenario for targeted retraining and testing.

5.3.3. Abstract knowledge scenarios for testing relevant knowledge infusion

This section explains how we construct TSCs for the Knowledge Infusion Test. The Relevant Knowledge Infusion Test checks whether the knowledge infusion of knowledge K_{req} into M_n was successful, i.e. whether M_n has grasped the requirement R and the knowledge requirement K_{req} that was intended to be infused. To test this, we (i) construct a knowledge TSC combining K_{desc} and R_{desc} from which concrete scenarios for a (ii) test suite are synthesized, which are then monitored to establish whether (iii) M_n behaves as required by K_{cons} and R_{cons} . Fig. 9 illustrates how the knowledge TSC is constructed. To construct the knowledge TSC, the requirement R_{desc} and knowledge K_{req} are composed via the *Concurrency*. In other words, R and K_{req} are composed, ignoring their consequences R_{cons} and K_{cons} . We thus construct a TSC that encodes scenarios where M_n is supposed to show the required behavior but does not limit the scenarios any further. It thus remains the obligation of M_n to display the required behavior to satisfy the conjunction of R_{cons} and K_{cons} . We use the synthesis approach of Becker et al. [48] to derive a concrete test scenario. We then use the monitoring approach of Grundt et al. [32] to check satisfaction of the requirement(s) consequence, i.e. R_{cons} and K_{cons} .

Note that the same construction can be used to synthesize (re-)training data for an identified knowledge gap. Moreover, for Reinforcement Learning, K_{cons} can be integrated into the reward function of the training by reward shaping. The realization of Reinforcement Learning based on TSCs is described in [14] and is used within the case studies presented in Section 7.

Example. Let us consider the systematic construction for *AI Training* and *AI Evaluation* (cf. Section 2) for our running example. We have the system requirements specified via TSCs as given in Fig. 6 on p. 14) and we have the potentially relevant knowledge as illustrated in Fig. 7 on p. 14. Our systematic derivation of an abstract knowledge scenario combining both specifications is illustrated in Fig. 10. We can use this TSC for targeted adaption of *AI Training* environments and targeted *AI Evaluation*. \square

5.3.4. Benefits of abstract knowledge scenarios

The presented abstract knowledge scenarios can be useful independent of our testing procedure. A targeted training hinges on a *good compilation of the training data set*. Therefore, the specification of the different knowledge scenarios of the training set is an important step. Provided we have a training set given and specified what abstract knowledge scenarios should be part of the training set, we can *examine the coverage of knowledge scenarios and how balanced the training data is* regarding the different knowledge scenarios. Moreover, when an initial AI model violates its requirements, testing whether the knowledge-infused AI model succeeds in the knowledge scenarios can now be *tested and monitored* easily using our abstract knowledge scenarios since a formalization forms the basis for automatable satisfiability checks. In addition, our formal abstract knowledge scenario specification can be checked for consistency. The formal basis also ensures a high level of tool interoperability. As shown in Fig. 13, for example, TSC specifications can be translated into the OpenX standard [51] to create the basis for selecting training and test environments or generating synthetic data. In addition, a formal specification helps communication between stakeholders as it has no room for multiple interpretations.

Therefore, by formally specifying requirements in combination with identified knowledge, we also address practitioners' challenges C1 and C2 (cf. Section 4.1).

6. Relevance testing procedure

In the context of knowledge-infused AI, infusing relevant knowledge aims to yield an AI driving function that generalizes better and can cope better with unknown situations. Thus, it is expected to satisfy requirements more reliably. Consequently, the training is also expected to be faster and more cost-effective than solely data-driven machine learning.

Unfortunately, the path from *Knowledge Identification* over *Knowledge Formalization* to *Knowledge Integration* is rather long (cf. Section 2). Current training and evaluation methods do not indicate whether the identification and formalization of infused knowledge were correct, nor do they provide any concrete information on whether the identified knowledge was relevant to satisfying the corresponding requirement. We see the identification and the correct formalization as essential factors for creating beneficial knowledge bases for AI driving functions and developing efficient and robust knowledge-infusion approaches.

We present a testing procedure that determines scenario-relevance indications (cf. Section 4.3.2, p. 10) of whether identified and formalized knowledge is relevant for the AI. It thereby also helps to answer the question “*Is the formalized knowledge beneficial for the AI driving function?*”. Scenario-relevance indications examine whether a given knowledge is relevant to the initial model M_i in the scenario Scen . Therefore, a test suite \mathcal{S} is executed.

6.1. Relevant knowledge infusion test procedure

For the reader's convenience, we refer to Table 1 (page 6) for the denotations used in the following. We moreover refer to Table 4 and page 13 where the knowledge specification is discussed in more detail.

For the following, we denote with M_i the initially trained AI model, which does not satisfy the requirement R .¹¹ The AI model M_i is trained further with data representing identified and formalized knowledge \mathbb{K} in order to make it satisfy the requirement R . We call the retrained model M_n .

The to-be-infused knowledge \mathbb{K} aims to fix the shortcomings of M_i . If M_n implements \mathbb{K} and does not forget what M_i already knew, then it should satisfy R . As discussed in the previous section on p. 13, we assume that the potentially relevant knowledge \mathbb{K} is specified as a requirement K_{req} and formalized via HiC-TSCs or HaFiC-TSCs. The Premise of K_{req} , also called K_{desc} , describes a subscenario for which the consequence K_{cons} is required from the AI model. In our running example $K_{\text{desc}} = \text{“It is rainy and the friction changes”}$, $K_{\text{cons}} = \text{“Brake safely and take the changed friction into account.”}$ and $K_{\text{req}} = \text{“If it is rainy and the friction changes, then brake safely and take the changed friction into account.”}$.

In the following, we use the more intuitive term “behavior” instead of the more formal term trajectory.

6.2. Relevant knowledge infusion test

In this section, we first give an overview of our testing procedure and then explain it in more detail using our running example.

Fig. 11 gives an overview of our Relevant Knowledge Infusion Test. Basically, we test relevance with the notion in mind: “ $M_i \not\models R$ and $M_i \odot \mathbb{K} \models R \Rightarrow \mathbb{K}$ is relevant”. That is, if M_n satisfies R , but M_n without \mathbb{K} , that is, M_i , does not satisfy R , we consider \mathbb{K} as relevant for R . Moreover, note that we usually cannot do exhaustive simulation, so that simulation will provide only relevant indications as discussed in Section 4.3 p. 10.

The Relevant Knowledge Infusion Test is to be used after the steps `training` and `infusion` have been performed, i.e. after the initial AI model M_i has been trained to satisfy R but does not satisfy R and the new AI model M_n has been trained by infusing \mathbb{K} into M_i .

In order to derive indications of whether knowledge \mathbb{K} is irrelevant, contradicting previous knowledge, or is partly relevant, we combine the outcomes of the three tests `Knowledge Infusion`, `Knowledge Preservation` and `Requirement Satisfaction`.

In the following, we use Table 5 to explain our testing procedure in more detail. We illustrate our testing procedure in Section 7. To simplify the discussion, we first pretend to simulate exhaustively; hence, in Table 5, we pretend to have set inclusion \subseteq rather than observed set inclusion $\subseteq_{\mathcal{S}}$. We then discuss what the selective execution of test cases in simulation means for our testing procedure.

In column “*K Infused*” of Table 5, it is listed whether the new model M_n behaves as required by \mathbb{K} ($M_n \subseteq \mathbb{K}$). This test checks whether M_n internalized the provided knowledge \mathbb{K} . Reasons for failing these tests may be the inadequacy of the training, the model itself or the knowledge.

Column “*M_i Preserved*” lists whether M_n preserves the successful behavior of the initial model M_i . To this end, we first determine the test runs where M_i satisfies R . We then check whether, in all these cases, M_n satisfies R . If this test fails, the knowledge infusion overwrote the knowledge K_i that M_i previously learned. Reasons may be that the model is not adequate (e.g., it may be too small to produce the complex behavior and hence forgets) or the training caused overfitting, or the initial knowledge K_i may contradict the infused knowledge (e.g., the infused knowledge labels cases as appropriate for comfortable breaking while it was previously labeled as hard breaking).

¹¹ In case there is no such model, we pretend that there is a model M_n that non-deterministically chooses its actions.

```

\* Initially given*\
    • R, the requirement
    • Mi, the trained AI model which does not satisfy R;
      Mi := Mn, if there is no initially trained model
    • knowledge K that will be infused

\* Knowledge Infusion Training*\
    • Mn = Mi ∘ K, i.e. Mi is further trained by knowledge infusion of K

\* Relevant Knowledge Infusion Test*\

1. Knowledge Infusion Test: Was K infused into Mn?
   % It checks whether the attempt to infuse knowledge K via training
   % into AI model Mn was successful. Formally, it is examined whether
   % Mn ⊆δ K, i.e. whether the behaviors of Mn satisfy the knowledge-
   % requirement K.

2. Knowledge Preservation Test: Are the successes of Mi preserved?
   % It checks Mn ≥δ,R Mi, i.e. whether the Mn did not forget initial knowledge Ki. Therefore
   % it checks, if Mn satisfies R in all test cases for where Mi satisfies R.

3. Requirement Satisfaction Test: Does Mn satisfy the requirement R?
   % It checks whether the Mn is successful in all test cases of requirement
   % R. Formally, it is examined whether Mn ⊆δ R, i.e. whether the
   % behaviors of Mn satisfy the requirement R.

return indications whether K is relevant/irrelevant/contradicting previous knowledge.
    
```

Fig. 11. Sketch of the overall Relevant Knowledge Infusion Test.

Table 5
Overview of indications provided by the Relevant Knowledge Infusion Testing Procedure.

K Infused?	M _i Preserved?	Req. R Satisfied?	Indication
$M_n \not\subseteq_{\delta} K$	$M_n \not\geq_{\delta,R} M_i$	$M_n \not\subseteq_{\delta} R$	K consistent? M _n adequate? training adequate?
$M_n \not\subseteq_{\delta} K$	$M_n \geq_{\delta,R} M_i$	$M_n \not\subseteq_{\delta} R$	K consistent? M _n adequate? training adequate?
$M_n \not\subseteq_{\delta} K$	$M_n \not\geq_{\delta,R} M_i$	$M_n \subseteq_{\delta} R$	not possible
$M_n \not\subseteq_{\delta} K$	$M_n \geq_{\delta,R} M_i$	$M_n \subseteq_{\delta} R$	K consistent (with K _i)?
$M_n \subseteq_{\delta} K$	$M_n \not\geq_{\delta,R} M_i$	$M_n \not\subseteq_{\delta} R$	K consistent (with K, R)? K partly relevant? K irrelevant?
$M_n \subseteq_{\delta} K$	$M_n \not\geq_{\delta,R} M_i$	$M_n \subseteq_{\delta} R$	not possible
$M_n \subseteq_{\delta} K$	$M_n \geq_{\delta,R} M_i$	$M_n \not\subseteq_{\delta} R$	K sufficient? K partly relevant? K irrelevant?
$M_n \subseteq_{\delta} K$	$M_n \geq_{\delta,R} M_i$	$M_n \subseteq_{\delta} R$	K scenario-relevant

Column “Req. R Satisfied?” lists whether M_n satisfies the requirement R. Possible reasons for failing this test (ignoring the previous tests) are inadequacy of the model, training process or data.

Combining the test results gives us stronger indications of the reasons for failing the three tests. We discuss the possible combinations one by one in the following.

Row 1: Suppose we established that M_n does not internalize the knowledge K, it does not preserve the successes of M_i, and it does not satisfy the requirement R. Since M_n forgot what it initially knew (before knowledge infusing M_n equals M_i) there is an indication that M_n did learn something. Hence, problems in the training data i.e. the consistency of knowledge, should be examined; the model might also be inadequate. Since M_n fails in cases where M_i was successful, it is less likely that more training is beneficial.

Row 2: Let us assume M_n does not internalize \mathcal{K} (i.e. does not satisfy \mathcal{K}) but preserves the successes of M_i and it does not satisfy the requirement \mathcal{R} . We have no indication that M_n learned something. Hence, the training data, the model, or the training might not be adequate. Additional information is needed to rule out some of these cases.

Row 3 and 6: These cases are listed for combinatorial completeness. They are not possible since if M_n fails in some cases according to the preservation test ($M_n \geq_{\mathcal{S}, \mathcal{R}} M_i$), then M_n cannot satisfy \mathcal{R} .

Row 4: Suppose we established M_n does not internalize \mathcal{K} , it preserves the successes of M_i and it satisfies the requirement \mathcal{R} . Since M_n learned to satisfy \mathcal{R} there is an indication that M_n did learn something. Hence, problems in the training data, i.e., the consistency of knowledge, should be examined. The infused knowledge itself might be contradictory (which would mean that M_n cannot satisfy \mathcal{K}), or contradicting previously learned knowledge \mathcal{K}_i . If e.g. EWC is used (cf. Section 3), and the latter becomes more likely.

Row 5: Suppose we established M_n internalizes \mathcal{K} , it does not preserve the successes of M_i and does not satisfy \mathcal{R} . Since M_n has forgotten parts of \mathcal{K}_i , there is an indication that M_n did learn, but what M_n learned is not enabling it to satisfy \mathcal{R} . Hence, problems in the training data, i.e., the consistency of knowledge, should be examined. The infused knowledge \mathcal{K} seems to contradict previously learned knowledge \mathcal{K}_i . Whether or not the infused knowledge brings an improvement needs further analysis. We hence examine whether M_n or M_i perform better wrt. satisfying \mathcal{R} . As a measure, we use the number of behaviors that satisfy \mathcal{R} . If M_n has more behaviors satisfying \mathcal{R} , (at least part of) \mathcal{K} might be relevant and \mathcal{K}_i might be inconsistent. The case “ $|M_i \cap \mathcal{R}| \not\leq |M_n \cap \mathcal{R}|$ ”, i.e. that M_i has not the right more behaviors satisfying \mathcal{R} , indicates that \mathcal{K} is not relevant.

Row 7: Suppose M_n internalizes \mathcal{K} , it preserves the successes of M_i and does not satisfy \mathcal{R} . Since M_n has learned \mathcal{K} there is an indication that M_n did learn, but what M_n learned is not enabling it to satisfy \mathcal{R} . There is no indication that knowledge is inconsistent. Hence, whether the infused knowledge \mathcal{K} is irrelevant or partly relevant should be examined. We hence examine whether M_n or M_i performs better wrt. satisfying the requirement \mathcal{R} in terms of the number of behaviors that satisfy \mathcal{R} . If M_n has more, \mathcal{K} might be relevant but insufficient, and further knowledge might be required. Note, that “ $|M_i \cap \mathcal{R}| \not\leq |M_n \cap \mathcal{R}|$ ” in combination with $M_n \geq_{\mathcal{S}, \mathcal{R}} M_i$, means $|M_i \cap \mathcal{R}| = |M_n \cap \mathcal{R}|$. The case indicates that \mathcal{K} is not relevant.

Row 8: Suppose M_n internalizes \mathcal{K} , it preserves the successes of M_i and satisfies \mathcal{R} . This case indicates that \mathcal{K} is scenario-relevant. Note that \mathcal{K} is sufficient does not imply that all of \mathcal{K} is necessary. There can be less specific knowledge \mathcal{K}' also conveying sufficient knowledge with $\mathcal{K} \subseteq \mathcal{K}'$.

Non-exhaustive exploration in simulation For practical reasons, a limited number of test cases are usually simulated. The coverage criteria and selection of test cases are part of the test design, which is done by test engineers. Criticality of the respective requirements, complexity of the system, and experience influence what test cases will be executed. To show that $\mathcal{M} \not\subseteq \mathcal{M}'$ a single counter-example suffices, while showing $\mathcal{M} \subseteq \mathcal{M}'$ would require exhaustive exploration of all elements of \mathcal{M} . This means that, e.g., in the row 5 with $M_n \subseteq_{\mathcal{S}} \mathcal{K}$ we could have $M_n \not\subseteq_{\mathcal{S}} \mathcal{K}$ as well and hence face the case of row 1. Our algorithm ignores this aspect, trusting on an appropriate coverage. The key to using relevance indicators is to be aware of this limit. Databases that maintain relevant knowledge should document the coverage criteria, and additional tests should be done if more evidence is needed.

Relevance notions While rows 1-4 indicate that \mathcal{K} is irrelevant for this trained model in this scenario, only rows 5-8 specify cases where indications can be derived that \mathcal{K} is partly relevant. Only in row 8 we can derive that \mathcal{K} is sufficient to satisfy the requirement \mathcal{R} , while in rows 5 and 7, part of \mathcal{K} may be partly relevant, i.e., there are indications that \mathcal{K} is improving the behavior of the AI model but that \mathcal{K} is not sufficient. Our testing procedure does not aim to determine the least required knowledge but leaves it open to the test engineers whether they want to determine the least knowledge. Since the coverage in simulation and the model-world gap induce uncertainties into the indicators, a systematic search for the lesser knowledge should be driven by domain knowledge.

7. Evaluation and application of relevance testing procedure

This section aims to demonstrate the overall approach (cf. Fig. 4), which supports research and industry addressing the question RQ (motivated in Section 1 and Section 4.1):

RQ: Is the infused knowledge relevant for an AI model to fulfill its task?

To this end, the presented examples illustrate how different types of knowledge for safety requirements are formalized, infused and evaluated regarding their relevance for the considered initial AI model. In addition, the selected examples show (i) that the evaluation of relevance provides additional insights into the infused knowledge, namely that the knowledge is relevant and not, e.g. purely statistically correlated, and (ii) how the relevance indications provide (causal) feedback as to why the contribution of relevant knowledge was not successful.

For the first example, we used a prototypical reinforcement learning framework developed for the TSC language [14]. The other two examples are theoretical and intended to demonstrate our approach’s versatility and generalizability.

7.1. Safe braking example

For the first example, we consider the requirement

R1 The system should maintain a safety distance of at least 2 m and up to maximum distance of 15 m from a static obstacle

and trained a Reinforcement Learning (RL) agent for R1.

7.1.1. Training specifications

For the training, we used a prototypical RL framework from [14]. We used a soft actor-critic algorithm, the Stable-Baseline3 [52] and Gymnasium [53] for the training environment. Vehicle dynamics in the environment are modeled using the well-known kinematic bicycle model [54]. The differential equations as in [14] were used for the dynamics of the agent system called *ego*:

$$\begin{aligned} \dot{x}_{ego} &= v_{ego} \cos(\theta_{ego}), & \dot{y}_{ego} &= v_{ego} \sin(\theta_{ego}), \\ \dot{\theta}_{ego} &= v_{ego} \frac{\tan(\delta_{ego})}{L}, & \dot{v}_{ego} &= a_{ego}, \end{aligned} \quad (1)$$

with a two-dimensional position for *ego* (x_{ego} , y_{ego}) and obstacle (x_{obs} , y_{obs}), a yaw heading θ_{ego} and a velocity v_{ego} (θ_{obs} and v_{obs} respectively). $L = 3.1$ m denotes *ego*'s wheelbase, while in the environment, the lane has a constant width of 4 m and a fixed position.

We enrich the kinematic model for the experiment by a friction coefficient (cf. Equation (2)).

$$\begin{aligned} F_N &= mass_{ego} \cdot g, \\ F_r &= \mu \cdot F_N, \\ a_{friction_{ego}} &= \frac{F_r}{mass_{ego}}, \\ a_{ego} &= a_{ego} + (-a_{friction_{ego}}), \\ \dot{v}_{ego} &= v_{ego} + a_{ego} \cdot \Delta t \end{aligned} \quad (2)$$

We first calculate the normal force F_N with the gravitational acceleration g and the mass of *ego* $mass_{ego}$. We use $mass_{ego} = 1700$ kg. We calculate the frictional force F_r with a friction coefficient μ and F_N . The frictional delay can be calculated with F_r and $mass_{ego}$. The effect of the friction coefficient μ on the deceleration a_{ego} can be calculated using the negated friction delay.

Since the agent selects a_{ego} in the action space, the velocity in the next time step \dot{v}_{ego} is calculated in the kinematic model as a function of a_{ego} , the current velocity v_{ego} and Δt . The friction coefficient μ is considered by our extension in a_{ego} and is therefore included in calculating the velocity at time $t + \Delta t$.

Observation and action space Given the requirement R1 and the respective TSC specification depicted earlier in Fig. 6, we assume an observation space of the agent, which only contains the distance between the agent system *ego* and the static obstacle *obstacle*, and the velocity of *ego*.

The action space of the agent contains the acceleration of *ego*. The acceleration of *ego*, a_{ego} , can range from -7 m/s² to 4 m/s². We assume that *obstacle* is positioned in *ego*'s lane. Due to the given observation space and actions space, we have set the input parameter for the steering angle δ_{ego} and θ_{ego} in Equation (1) constant in the middle of the lane and straight in the direction of *obstacle* for the training.

Reward function We perform reward shaping for requirement R1 with two terms for the first training session:

$$\begin{aligned} x_{obs} - x_{ego} &> 2 \text{ m}, \\ x_{obs} - x_{ego} &< \max(15 \text{ m}, v_{ego}). \end{aligned} \quad (3)$$

The first term enforces a safety distance of 2 m to be maintained. The second term states that the agent should maintain a maximum distance equal to v_{ego} at high speed or a maximum of 15 m. When these terms are taken into account, an appropriate safety distance to a stationary obstacle can be maintained without *ego* being able to stand any distance away for a high reward and at the same time maintain the safety distance of 2 m. Within the specified lower and upper limits for the distance, the reward (between 0 and 1) is calculated as a function of *ego*'s speed using the modified sigmoid function Equation (4).

$$\text{reward}(f) = c \cdot \left(\frac{1}{1 + e^{b \cdot (f+a)}} \right) + d \quad (4)$$

The constant parameters a , b , c , and d can be chosen according to the desired distribution of the reward, we used $a = 10$, $b = 0.25$, $c = 1$, and $d = 0$. Here, a determines the distance to the inflection point, b determines the function slope, c determines the maximum reward and d is the minimum reward. The function f represents the value evaluated in each training step according to Equation (3).

Training setting The *ego* agent is trained in episodes with distances ranging from 70 m to 100 m to *obstacle*, and starting speeds between 10 m/s to 30 m/s. The agent is trained in 1 million steps, approximately 4000 episodes, with a learning rate of 0.02.

7.1.2. Example 1 - verification of M_i

After training, the agent was tested regarding requirement R1. The results show it does not satisfy R1 in all tests (cf. Section 4.3.2), i.e. M_i does not satisfy R in Scen vbt .

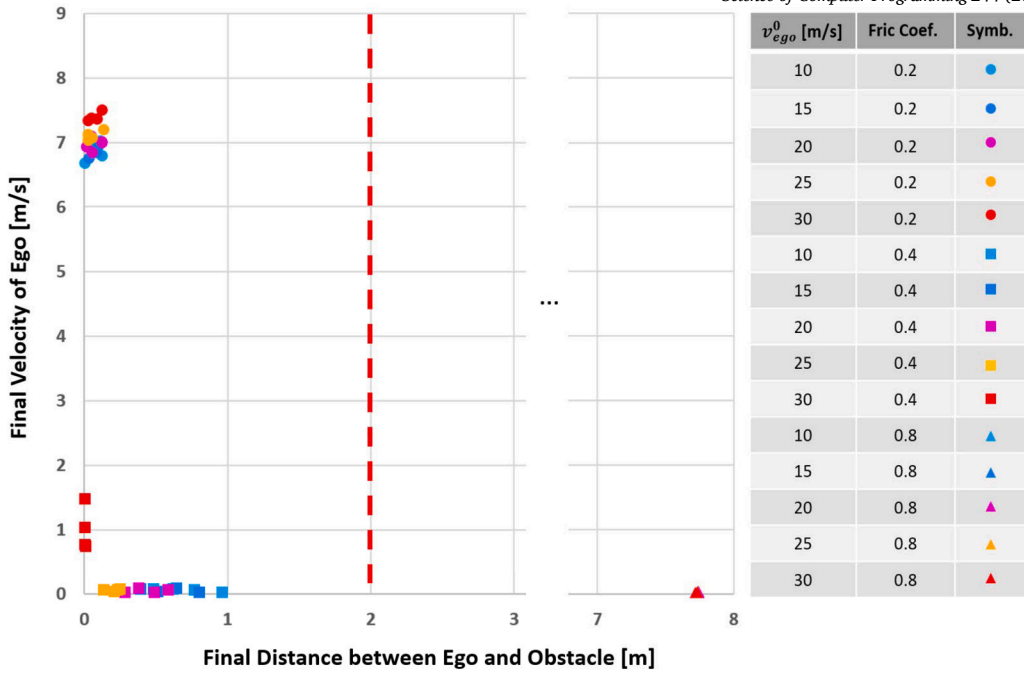


Fig. 12. Test results of a trained agent M_i . The agent was able to satisfy requirement R1 in test runs with a prevailing friction of 0.8 - dry road (triangles). In test runs, in which the prevailing friction is 0.4 - wet road (squares) or 0.2 - icy road (circles), the agent violates R1. (For interpretation of the colors in the figure(s), the reader is referred to the web version of this article.)

During testing, an environment was chosen that specifies a prevailing friction using a friction coefficient. The agent has been tested in environments with the friction coefficients 0.8 - dry asphalt, 0.4 - wet asphalt (i.e., ≥ 1.5 mm surface water), and 0.2 - icy roads.

The test results are presented in Fig. 12. The distance between the *ego* and *obstacle* is on the x-axis, and *ego*'s velocity is on the y-axis. Each point encodes a vector of final values of a single test run in term of *ego*'s speed, the distance between *ego* and *obstacle*. The color of a point encodes *ego*'s initial velocity: red 30 m/s, orange 25 m/s, magenta 20 m/s, dark blue 15 m/s, and light blue 10 m/s. The agent was also tested with the same start distances to the *obstacle* as in training. This is not depicted separately.

Fig. 12 show that the initially trained agent does not comply with the safety distance of 2 m, if the friction coefficient changes in the environment, i.e. the trained agent collides with the obstacle on wet and icy roads.

7.1.3. Specification of abstract knowledge scenario

Based on the verification result of the initially trained agent M_i , it can be concluded that scenario-relevant knowledge about a possible change in the friction coefficient and the associated necessary adaptation of braking distances could lead to satisfaction of requirement R1. As a specification of this knowledge in relation to requirement R1, we use the earlier specified TSC specification depicted in Fig. 7.

Combination of requirement and knowledge With the derivation of abstract knowledge scenarios presented in Section 5, we create a formal foundation for the identification, integration, sharing, and archiving of scenario-relevant knowledge. With this, we address the challenges of practitioners (cf. Section 4.1) with the benefits discussed in Section 5.3.4. Given the identified knowledge and the requirement R1 as a TSC specification, we can combine the specifications as previously presented. The result is depicted in Fig. 10.

Based on the combined specification, concrete and simulatable scenarios can be derived using suitable tooling. As a result, e.g., in Reinforcement Learning (RL), the identified knowledge gap can systematically expand the training environment. A tool for generating concrete scenarios for simulation already exists for the TSC language [48]. Furthermore, we used a prototypical RL framework for TSC specifications [14]. Considering Reinforcement Learning and testing, the consequences of the requirement specification and knowledge specification shall be satisfied by an RL agent and, hence, should not be covered in the derived abstract knowledge scenario. Our experiment uses the TSC consequences as the basis for an existing online monitoring of TSC specifications [32]. This enables an efficient execution of the testing procedure presented in Section 6. Hence, it enables the execution of the defined Knowledge Infusion Test and Requirement Satisfaction Test in each training and test run during execution. Furthermore, the Knowledge Preservation Test can be performed directly after executing all training and test runs. The mentioned toolchain is depicted in Fig. 13.

Concretely, online monitors [32] can be synthesized based on TSC consequences. By applying our derivation and a resulting abstract knowledge scenario, concrete scenarios can be derived and simulated in, e.g., CARLA [50] using the TSC2CARLA tool chain [46]. The knowledge-infused agent can be integrated into the simulation environment, i.e., a vehicle (CARLA actor), and

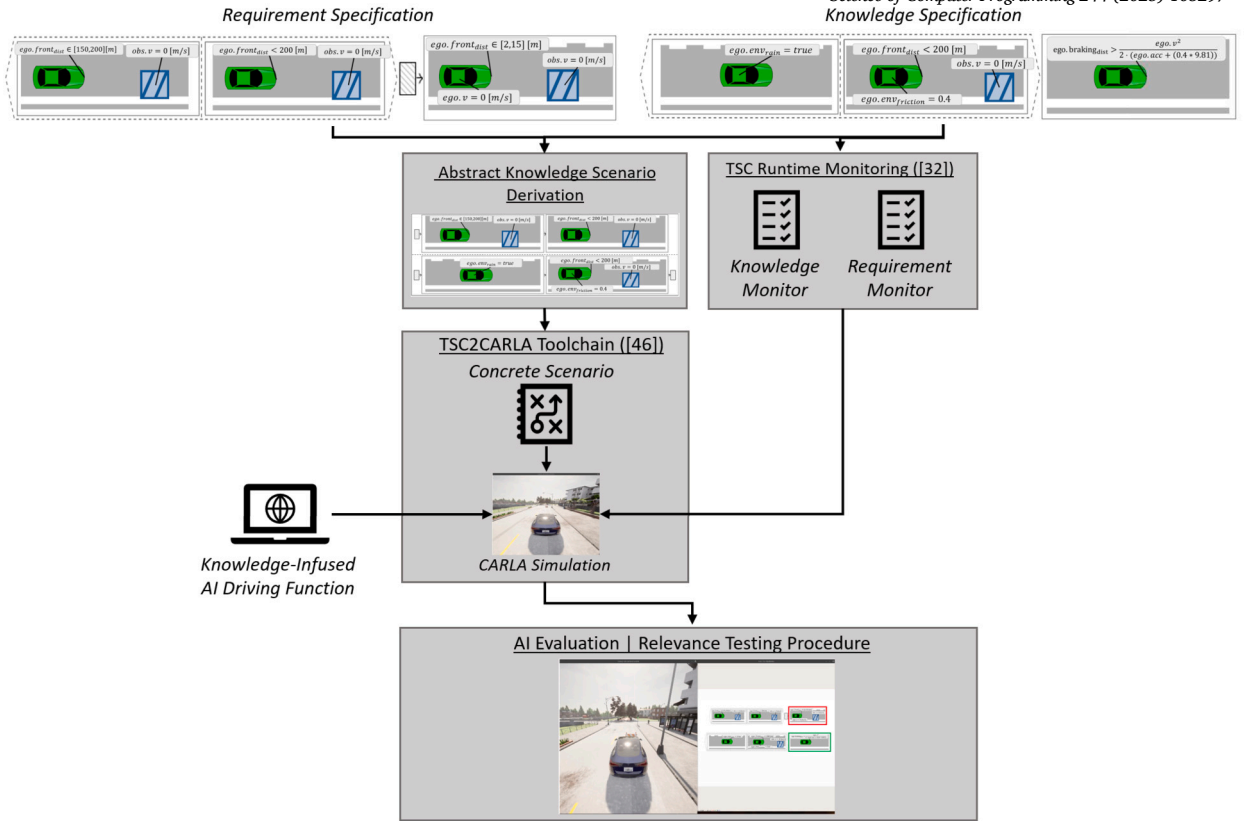


Fig. 13. Pipeline of relevance testing procedure incorporating TSC Runtime Monitoring [32] and TSC2CARLA toolchain [46].

executed in a concrete scenario simulation. Consequently, the monitors can provide verdicts on the knowledge conformance and requirement satisfaction at runtime, i.e., the input for the presented relevance testing procedure.

7.1.4. Training and verification of M_n

Given the combined specification of requirement R1 and the identified knowledge, we adapted the training environment. First, we used different but fixed friction coefficients in the training scenarios. Furthermore, we extended the observation space of the agent by the prevailing friction coefficient. Finally, we trained an agent M_n with the same learning rate, number of training steps, and episodes as before.

In the verification process, we performed the relevance testing procedure presented outlined in Fig. 13. We still used the Gymnasium environment as training and test environment.

Based on our abstract knowledge scenario derivation, we were able to obtain monitors for all three tests *Knowledge Infusion Test*, *Knowledge Preservation Test*, and *Requirement Satisfaction Test* and run them in each test run.

Example 2 - inadequate knowledge-infused training In the following, we show a knowledge-infused M_n where the \odot has already been applied, but inadequate training is present (less than 500,000 training steps). The result of each test run of this model is depicted in Fig. 14. The results of the relevance testing procedure are as follows: After all test runs, the Knowledge Infusion Test gives the result $M_n \not\subseteq_S \mathbb{K}$. The Knowledge Preservation Test yields $M_n \geq_{S,R} M_i$. Finally, the Requirement Satisfaction Test yields $M_n \not\subseteq_S \mathbb{R}$. This result combination is reflected in row 2 of Table 5.

Given these results, we check the indications obtained. We can rule out that \mathbb{K} is inconsistent, as the Knowledge Preservation Test was successful and $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$. Thus, M_n has learned to satisfy \mathbb{R} in more test runs than M_i . We can also rule out the possibility that, for example, the architecture of M_n is not able to learn the infused knowledge since the knowledge to be considered is already successfully applied in some test runs.

Hence, we conclude that M_n seems to need more training steps.

Example 3 - scenario-relevant knowledge-infusion Given the indication for more training, the following is the verification result of the knowledge-infused M_n with more training steps. The results are depicted in Fig. 15.

The results display that for the Knowledge Infusion Test, the result is that the knowledge is successfully infused (cf. bottom half of Table 5). An important aspect is that the knowledge-infused agent is checked in the same tests as the M_i . Therefore, we were also able to perform the Knowledge Preservation Test in each test run and show that the success of M_i and M_n with less training steps is

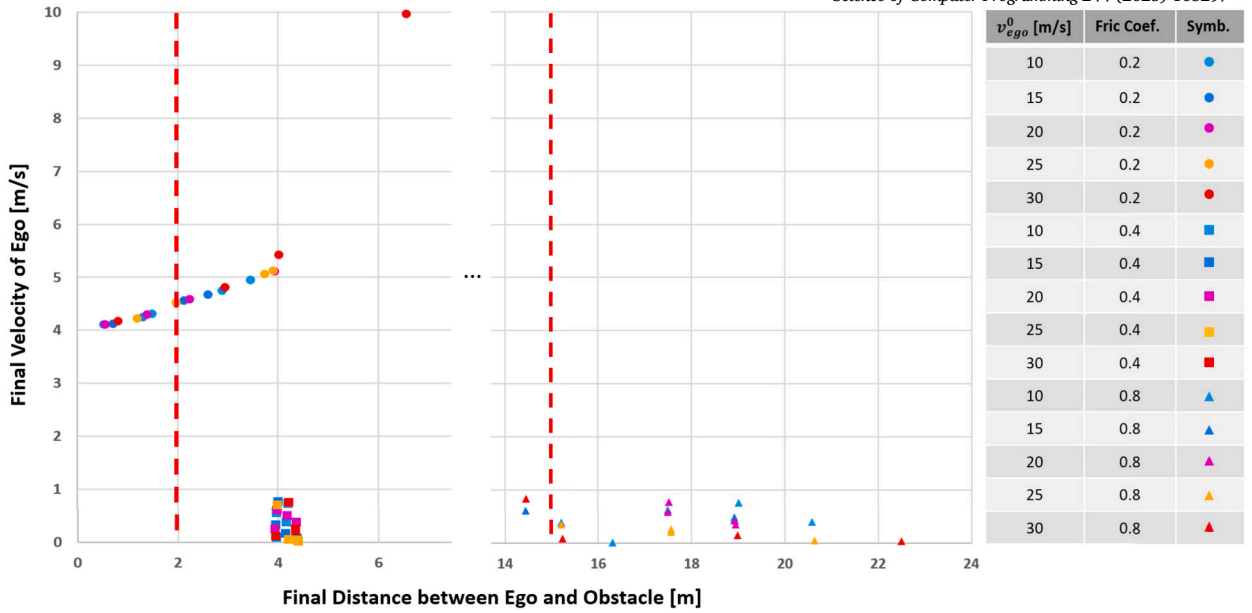


Fig. 14. Test results of knowledge-infused agent M_n . Compared to the first verification with M_i , the agent M_n could not satisfy requirement R1 in all test runs with a prevailing friction of 0.8 - dry road (triangles). However, the agent M_i satisfied R1 in all test runs with a prevailing friction of 0.4 - wet road (squares). In addition, M_n satisfies R1 in more than half of the test runs with a prevailing friction of 0.2 - icy road (circles).

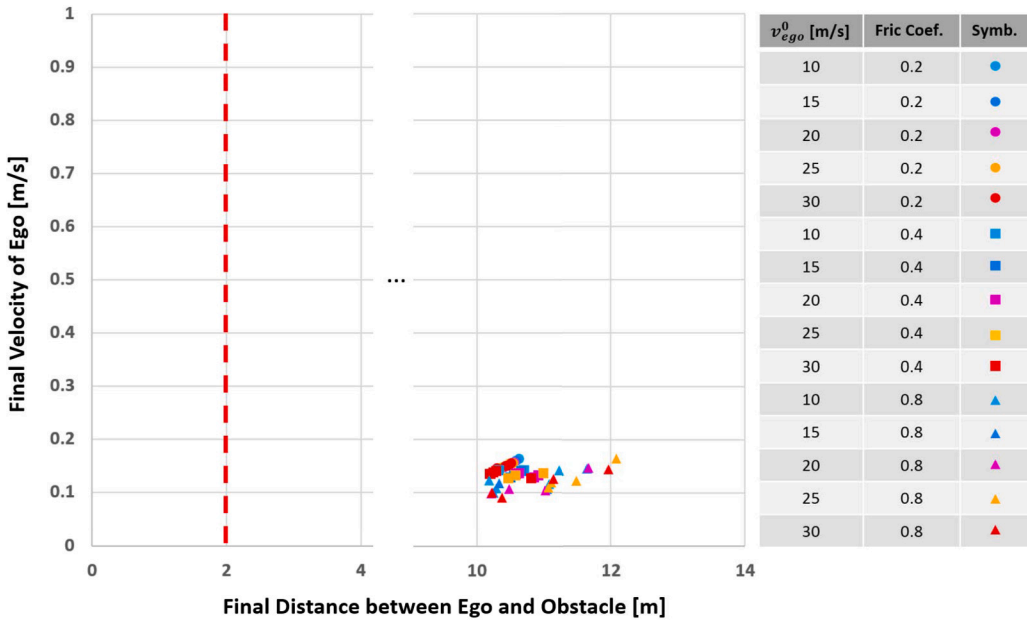


Fig. 15. Test results of knowledge-infused agent M_n with 1 million training steps. The agent M_n satisfies requirement R1 in all test runs, including the three different prevailing frictions of 0.8 - dry road (triangles), 0.4 - wet road (squares), and 0.2 - icy road (circles).

preserved by M_n (cf. last two rows of Table 5). At the same time, we also performed the Requirement Satisfaction Test, with the result that M_n satisfies R in every test run (cf. last row of Table 5) including all prevailing frictions.

In conclusion, the second verification shows that the Knowledge Infusion Operation \odot was successful and that M_n satisfies the requirement R1 in Scen vbt . Since we can show $M_n \subseteq_{\mathcal{S}} \mathbb{K}$, $M_n \geq_{\mathcal{S},R} M_i$ and $M_n \subseteq_{\mathcal{S}} \mathbb{R}$, we get an indication that \mathbb{K} is scenario-relevant.

Indications for other cases If the knowledge infusion had not been successful, we had to check the upper half of Table 5, for example, whether \mathbb{K} is consistent at all, if M_n has an inadequate architecture.

If the knowledge infusion were not successful and the result of the Knowledge Preservation Test were $M_n \not\geq_{\mathcal{S},R} M_i$, we would receive the indication that \mathbb{K} seems to be inconsistent. In the other case, M_n 's architecture seems to be inadequate for infusing \mathbb{K} .

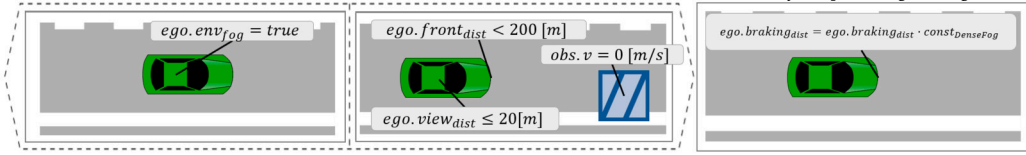


Fig. 16. TSC-specification of the correlation between dense fog resulting in short view distance and a change in the braking distance.

Given a successful knowledge infusion, if the result of the Knowledge Preservation Test is $M_n \not\geq_{\mathcal{S},R} M_i$, we get indications that \mathbb{K} is partly relevant if $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$ or that \mathbb{K} is irrelevant if $|M_i \cap \mathbb{R}| \geq |M_n \cap \mathbb{R}|$. If $M_n \not\subseteq_{\mathcal{S}} \mathbb{R}$, we get the indication that \mathbb{K} is inconsistent.

Lastly, given a successful knowledge infusion and the result of the Knowledge Preservation Test is $M_n \geq_{\mathcal{S},R} M_i$, we get indications that \mathbb{K} is partly relevant if $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$ or that \mathbb{K} seems irrelevant if $|M_i \cap \mathbb{R}| \geq |M_n \cap \mathbb{R}|$. If $M_n \not\subseteq_{\mathcal{S}} \mathbb{R}$, we get the indication that \mathbb{K} seems to be insufficient.

Finally, by evaluating this example, we demonstrated the feasibility of our proposed relevance notion, TSC-based knowledge scenario derivation, and the proposed relevance testing procedure to identify scenario-relevant knowledge systematically and formally. Hence, these methods can help practitioners address the question of whether infused knowledge is relevant for an AI model to satisfy its task.

7.2. Relevance vs. Statistical correlation

In the previous experiment, we show that we are able to obtain relevance indicators with our relevance testing procedure. Finally, we obtained the indication that knowledge on the extension of the braking distance due to the present friction is scenario-relevant. In order to show that relevance assessment for AI driving functions is important and differs from a common practice - using statistical correlations from, e.g. accident databases and statistics [55–58], we show two concrete examples.

The goal of this is to show that it is not sufficient to assume that statistical correlations are scenario-relevant. Without this indication, the potential benefits for i) validating early Knowledge-infusion phases, ii) generating a sustainable knowledge base and iii) supporting the verification and development of Explainable AI methods are not available.

7.2.1. Dense fog, visibility and braking distance correlation

Let us consider the correlation of accident statistics that rear-end collisions are often accompanied by bad visibility [59]. Bad visibility can occur, for example, due to snowfall or dense fog. We assume that the initially trained agent M_i does not satisfy the requirement in test runs with snowfall or bad visibility. In that case, this can lead to the mentioned correlation being considered as relevant knowledge (and not the prevailing friction). The correlation that the braking distance changes in bad visibility could be specified as a TSC as shown in Fig. 16:

Based on this specification and our derivation of abstract knowledge scenarios for training and testing purposes (composing R1 specification and this knowledge specification without the TSC consequence, cf. Fig. 9), we trained another agent using reinforcement learning. This time, the agent does not receive friction but the current visibility as an observable variable.

The agent was trained with the same reward function and training parameters as before (see Section 7.1.2). The new observation parameter split the same test runs as before (see Section 7.1.2 into test runs with good visibility (visibility = 100 m) and bad visibility (visibility = 20 m). The test results are shown in Fig. 17.

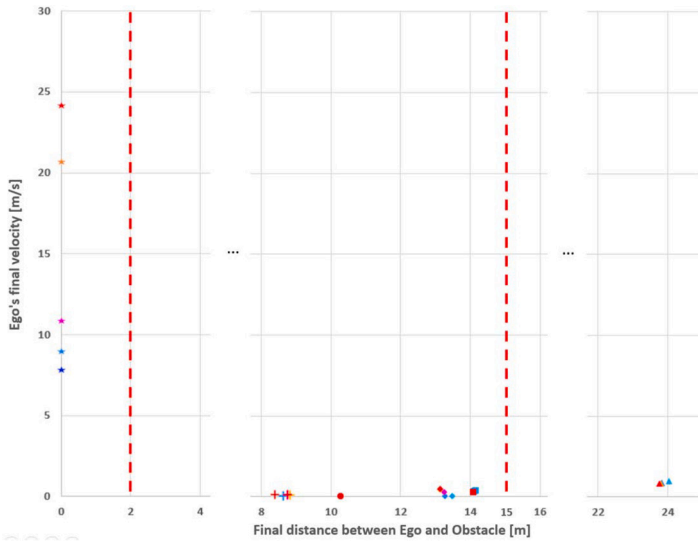
The results of the relevance testing procedure are as follows. After all test runs, the agent is able to satisfy the requirement on icy roads with bad visibility (circles), as well as on dry roads with good visibility (diamonds). Thus, the knowledge was successfully integrated and the Knowledge Infusion Test gives the result $M_n \subseteq_{\mathcal{S}} \mathbb{K}$. The Knowledge Preservation Test yields $M_n \geq_{\mathcal{S},R} M_i$. Finally, the Requirement Satisfaction Test yields $M_n \not\subseteq_{\mathcal{S}} \mathbb{R}$. This result combination is reflected in row 7 of Table 5. We therefore check the indications obtained. We can rule out that \mathbb{K} is irrelevant, as the knowledge preservation test was successful and $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$. Thus, M_n has learned to satisfy \mathbb{R} in more test runs than M_i . Next, \mathbb{K} seems not sufficient since $M_n \not\subseteq_{\mathcal{S}} \mathbb{R}$. However, since $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$, we get the indication that \mathbb{K} is partly-relevant.

The tests also show that the agent is not able to satisfy the requirement in the cases, which are not explicitly or implicitly specified by the correlation but plausible combinations: icy road - good visibility (stars), and dry road - bad visibility (triangles). Due to the same training duration as the agent in Fig. 15 and violation of the requirement R1 in specific test cases, we are able to conclude that the correlation remains partly-relevant and knowledge is missing.

7.2.2. Snow, average speed and braking distance correlation

Let us consider the correlation that rear-end collisions are often associated with high environmental speed in bad weather conditions [60]. This correlation specified as TSC is shown in Fig. 18:

Based on this specification and our derivation of abstract knowledge scenarios for training and testing purposes (composing R1 specification and this knowledge specification without the TSC consequence, cf. Fig. 9), we trained another agent using reinforcement learning. This time, the agent does not receive friction but only the surrounding speed of other vehicles as an observable variable. The agent was trained with the same reward function and training parameters as the final agent before (see Section 7.1.2). The new



v_{ego}^0 [m/s]	Fric Coef.	View [m]	Symb.	v_{ego}^0 [m/s]	Fric Coef.	View [m]	Symb.
10	0.2	20	●	10	0.2	100	★
15	0.2	20	●	15	0.2	100	★
20	0.2	20	●	20	0.2	100	★
25	0.2	20	●	25	0.2	100	★
30	0.2	20	●	30	0.2	100	★
10	0.4	20	■	10	0.4	100	+
15	0.4	20	■	15	0.4	100	+
20	0.4	20	■	20	0.4	100	+
25	0.4	20	■	25	0.4	100	+
30	0.4	20	■	30	0.4	100	+
10	0.8	20	▲	10	0.8	100	◆
15	0.8	20	▲	15	0.8	100	◆
20	0.8	20	▲	20	0.8	100	◆
25	0.8	20	▲	25	0.8	100	◆
30	0.8	20	▲	30	0.8	100	◆

Fig. 17. Test results of knowledge-infused agent M_n with partly-relevant correlation (visibility). The agent M_n satisfies requirement R1 in test runs with icy road and bad visibility (circles). This is the combination that the correlation expresses directly. The agent also satisfies R1 in test runs with dry road and good visibility (diamonds), which is just the indirect case of the correlation. The problematic cases are the ones which are not explicitly or implicitly covered by the correlation. The agent does not satisfy R1 in any test cases with icy roads and good visibility (stars) or dry roads and bad visibility (triangles). Additionally, the agent satisfies R1 in all test runs with wet roads. Here, the agent was successful by applying the same driving strategy as for icy roads and bad visibility.

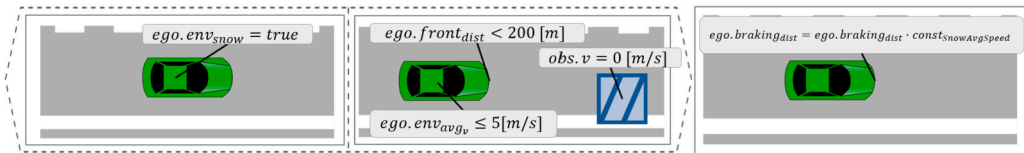
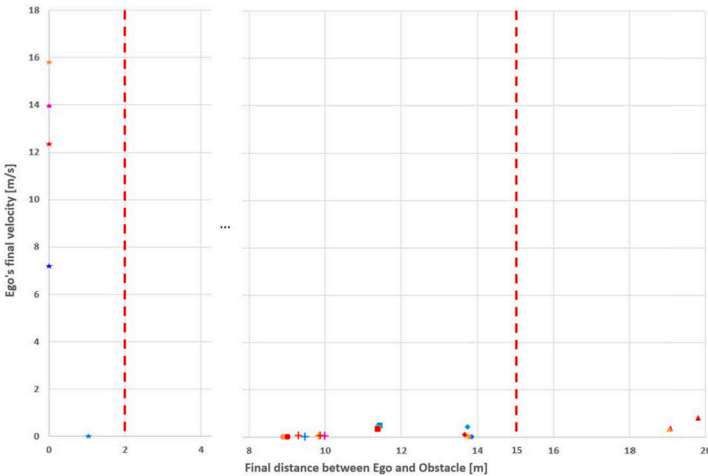


Fig. 18. TSC-specification of the correlation between snow, resulting in low average speed of surrounding traffic and a change in the braking distance.



v_{ego}^0 [m/s]	Fric Coef.	v_{avg} [m/s]	Symb.	v_{ego}^0 [m/s]	Fric Coef.	v_{avg} [m/s]	Symb.
10	0.2	5	●	10	0.2	27	★
15	0.2	5	●	15	0.2	27	★
20	0.2	5	●	20	0.2	27	★
25	0.2	5	●	25	0.2	27	★
30	0.2	5	●	30	0.2	27	★
10	0.4	5	■	10	0.4	27	+
15	0.4	5	■	15	0.4	27	+
20	0.4	5	■	20	0.4	27	+
25	0.4	5	■	25	0.4	27	+
30	0.4	5	■	30	0.4	27	+
10	0.8	5	▲	10	0.8	27	◆
15	0.8	5	▲	15	0.8	27	◆
20	0.8	5	▲	20	0.8	27	◆
25	0.8	5	▲	25	0.8	27	◆
30	0.8	5	▲	30	0.8	27	◆

Fig. 19. Test results of knowledge-infused agent M_n with partly-relevant correlation (surrounding speed). The agent M_n satisfies requirement R1 in test runs with icy road and low average speed of surrounding traffic (circles). This is the combination that the correlation expresses directly. The agent also satisfies R1 in test runs with dry road and high average speed of surrounding traffic (diamonds), which is just the indirect case of the correlation. The problematic cases are the ones which are not explicitly or implicitly covered by the correlation. The agent does not satisfy R1 in any test cases with icy roads and high average speed of surrounding traffic (stars) or dry roads and low average speed of surrounding traffic (triangles). Additionally, the agent satisfies R1 in all test runs with wet roads. Here, the agent was successful by applying the same driving strategy as for icy roads and low average speed of surrounding traffic.

observation parameter split the same test runs as before (see Section 7.1.2 into test runs with low average speed (5 m/s) and high average speed (27 m/s) of surrounding vehicles. The test results are shown in Fig. 19.

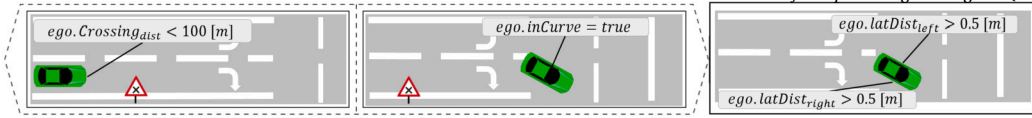


Fig. 20. TSC-specification of our exemplary requirement R2 “The system should always maintain a lateral distance of at least 0.5 m while turning”.

Given these tests, the relevance testing procedure yields the same results as for the correlation before: The agent is able to satisfy the requirement on icy roads with low average surrounding speed (circles), as well as on dry roads with high average surrounding speed (diamonds). Thus, the knowledge was successfully integrated and the Knowledge Infusion Test gives the result $M_n \subseteq_S \mathbb{K}$. The Knowledge Preservation Test yields $M_n \geq_{S,R} M_i$. Finally, the Requirement Satisfaction Test yields $M_n \not\subseteq_S \mathbb{R}$. This result combination is reflected in row 7 of Table 5. We therefore check the indications obtained. We can rule out that κ is irrelevant, as the knowledge preservation test was successful and $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$. Thus, M_n has learned something that satisfies \mathbb{R} in more test runs than M_i . Next, κ seems not sufficient since $M_n \not\subseteq_S \mathbb{R}$. However, since $|M_i \cap \mathbb{R}| < |M_n \cap \mathbb{R}|$, we get the indication that κ is partly-relevant.

The tests also show that the agent is not able to satisfy the requirement in the non-specified combinations: icy road and high average surrounding speed (stars), and dry road and low average surrounding speed (triangles). Due to the same training duration as the agent in Fig. 15 and violation of the requirement in specific test cases, which explicitly does not include the correlation, we are able to conclude that the correlation remains partly relevant and further knowledge is missing.

Note, that the agent generalizes in tests with wet roads (friction 0.4, not seen in training) and can satisfy the requirement using the same driving style as for icy roads and bad visibility. With other and more complex environmental properties, it may be the case that the knowledge preservation test results in $|M_i \subseteq \mathbb{R}| \not\prec |M_n \subseteq \mathbb{R}|$, then κ may even be irrelevant (see row 6 of Table 5). This can happen, for instance, if only a small part of property combinations given by the correlation is covered in tests and a majority of results do not satisfy the requirement. It can occur, for example, with correlations that are generally weak or unimportant in the specifically selected test runs.

In the presented examples, enriching training data by identified statistical correlations leads to better results. However, these correlations do not necessarily result from causal relations. Hence, they may lead to misguidance of the AI, as illustrated by the observation of the surrounding average velocity of other traffic participants and its correlation to the necessary braking distance. This shows that the pure use of statistical correlations may lead to better results, but the correlations are not equally relevant to the scenario. Therefore, compared to statistical correlations, only relevance indications provide more insights and increase the means for i) validating early Knowledge-infusion phases, ii) generating a knowledge base and iii) developing Explainable AI methods.

7.3. Safe turning - environmental knowledge specification

In the following, we introduce another example, which we will treat theoretically. We apply our methods for the second form of abstract knowledge specification *environmental descriptive* (cf. Section 5.2), in order to show feasibility.

Let us assume the requirement

R2 The system should always maintain a lateral distance of at least 0.5 m while turning.

Requirement R2 TSC-specification The TSC specification of R2 is depicted in Fig. 20. As before, we use a HaFiC-TSC and specify that if *ego* (green car) has approached a T-junction and turns, that *ego* should keep a lateral distance of 0.5 meter while turning.

Knowledge TSC-specification Let us assume that a trained agent does not satisfy R2 in all test runs, i.e., M_i does not satisfy \mathbb{R} . In the test runs, the agent could not maintain the required lateral distance due to weather-related environmental changes and sometimes swerved with the tail. Furthermore, this behavior occurs on wet and icy surfaces. In contrast to our first experiment, let us assume that we cannot obtain information about the prevailing friction. Therefore, we demonstrate how to utilize temporal properties about specific *environmental descriptive* knowledge using TSCs. Specifically, we utilize the current outside temperature and weather information.

Further assume that by analyzing test runs, it is recognized that the lateral velocity in turns is not adjusted depending on the current surface. As a result, the agent did not satisfy R2 in test runs with wet and icy surfaces. The maximum lateral acceleration in such weather conditions was identified as potentially scenario-relevant knowledge. The *environmental descriptive* specification for this knowledge is depicted in Fig. 21.

Combination of requirement and knowledge Based on the TSC specification of requirement R2 and potentially scenario-relevant knowledge, we can perform our abstract knowledge scenario derivation once again. In doing so, we provide a formal basis for closing the identified knowledge gap and for systematic integration and testing of the potentially scenario-relevant knowledge. With that, we address the challenges of practitioners (cf. Section 4.1) with the benefits discussed in Section 5.3.4.

Applying our derivation results in the TSC specification depicted in Fig. 22. Using the derived abstract knowledge scenario, we thus systematically offer a formal basis for targeted retraining and testing of, e.g., an RL agent specifically in the identified knowledge gap. Furthermore, based on the TSC specification and the presented tooling, we can again perform the three tests of the relevance testing procedure for each test run. In this example, the knowledge infusion test would check whether M_n adjusts the lateral acceleration under specified weather conditions. In this example, the Knowledge Preservation Test would check whether M_i maintains the lateral

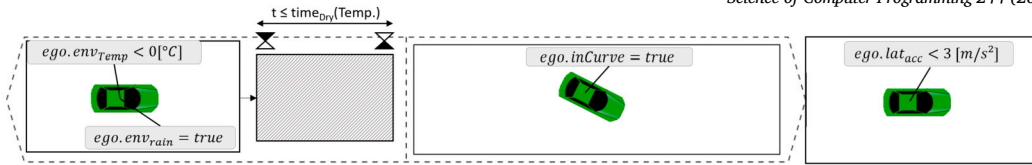


Fig. 21. TSC specification of our identified knowledge “The maximum lateral acceleration on icy roads should not exceed 3 m/s^2 . If weather conditions indicate icy roads, do not exceed a maximum lateral acceleration of 3 m/s^2 while turning.”. Given a preceding abstract scenario phase (history), where it rained, and the environment temperature was below 0°C . In addition, less time has passed than needed for the road to be free of ice or dry. Consequently, given by the identified knowledge, while ego is turning (future), do not exceed a maximum lateral acceleration of 3 m/s^2 (consequence).

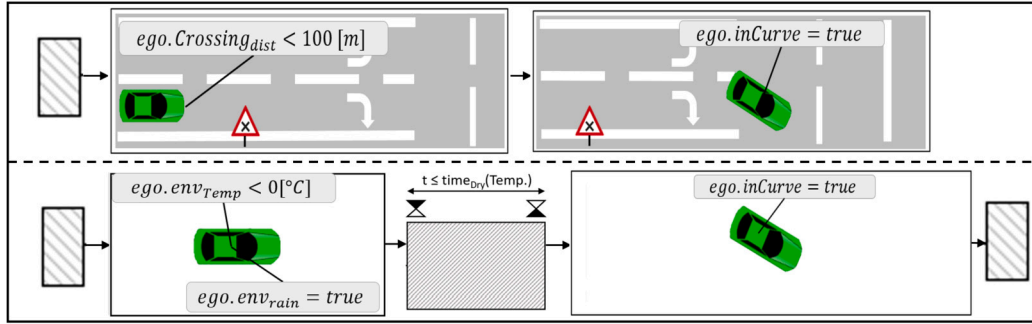


Fig. 22. Result of abstract knowledge scenario derivation combining requirement R2 and potentially scenario-relevant knowledge.

distance of 0.5 m in the test cases in which M_i was already able to do so. Finally, the Requirement Satisfaction Test would check whether M_n maintains the lateral distance in each test run of Scen . The respective indications of any combination of test results can be found in Table 5. In conclusion, with this example we presented and theoretically discussed the feasibility of our abstract knowledge specification with *environmental descriptive* knowledge.

7.4. Summary

In this case study, we used two examples to show how the presented methods can be combined and present our methods’ unique values. We demonstrated the systematic derivation of abstract knowledge scenarios using Traffic Sequence Charts for both examples. Concretely, we have specified both types of abstract knowledge scenarios $\mathcal{K}_{\text{desc}}$ and \mathcal{K}_{req} . Furthermore, we have executed the relevance testing procedure and showed that our method can i) provide several indications and ii) differentiate between causal relations and statistical correlation. For the first example, we trained a Reinforcement Learning agent into which knowledge was integrated by reward shaping. We used our methods to show that the identified and integrated knowledge is scenario-relevant according to our definition. Additionally, we trained two new agents and infused knowledge based on statistical correlation, which can be identified as potentially scenario-relevant for the first considered requirement. Our relevance testing procedure indicated that the statistical correlation is only partly relevant compared to the scenario-relevant knowledge about prevailing friction. Taking into account the assumptions for the specification of \mathcal{R} and \mathcal{K}_{req} (see Section 5), we can check all spatio-temporal properties that are specified as TSC and are observable for relevance to the performance of a knowledge-infused AI about the satisfaction of requirements. For the second example, we provided specifications of \mathcal{R} and $\mathcal{K}_{\text{desc}}$ and discussed the application of the relevance testing procedure theoretically. Overall, we demonstrated that our methods can support practitioners in answering whether infused knowledge is relevant for an AI model to satisfy its task. Based on these methods, we help address ML practitioners’ current challenges (cf. Section 4.1).

8. Related work

In the following, we present and discuss related work in knowledge-infused AI, specification of traffic scenarios, and scenario-based testing, including monitoring.

8.1. Knowledge-infused AI

Infusing prior knowledge into an AI was already realized by Towell et al. [26] in 1994. In its work, propositional logic rules were infused into a feed-forward network. With the revival of AI in the following years, the newly developed neural network types such as Bayesian [61] or support vector machines [62] were also investigated to determine how mathematical and physical knowledge can be infused. AI’s tasks became increasingly diverse due to technical and methodological developments. This led to the desire to infuse more complex mathematical and physical knowledge into an AI, e.g., for non-linear classifications [63]. In 2015, Reich et al. integrated mathematical and physical knowledge into Bayesian networks for forecasting and predictions in their work [64].

In digitalization and Industry 4.0, AI has also been used in manufacturing and attempts to infuse physical and mathematical prior knowledge, e.g., infusing monotonicity constraints using reward shaping [16].

Due to the rapid development of new network types and training methods, the research field of knowledge-infused AI has also focused on suitable solutions for infusing prior knowledge. Here, solutions exist, e.g., for auto-encoders [65], convolutional networks [66] and generative models [67], as well as for the training methods Reinforcement Learning [13,68] and Active Learning [18]. The field and the term physics-guided AI have been established based on these developments and experiences gained in infusing mathematical and physical knowledge. This includes all knowledge-infused AI approaches that deal with physical and mathematical knowledge.

Nowadays, the strengths of AI should also be utilized for highly automated and autonomous driving. In addition to physical and mathematical knowledge, handing over essential driving functions and decisions to an AI requires knowledge of traffic rules, social and societal norms, and ethics in the interaction between the real environment and public transport. Unfortunately, such knowledge is often not formalized. Therefore, a current challenge in knowledge-infused AI driving functions is the formalization of knowledge that is present in natural language. The works from Collette et al. [69], Borges et al. [34], Westhofen et al. [33] and also the research project KI Wissen [7] have dealt with how traffic rules can be formalized so that they can be used for the development of such systems.

To use AI driving functions in public transport, a certain level of acceptance and trust in such systems must also be achieved [8]. On the one hand, various studies are dealing with a suitable conformity test for verifying knowledge-infused AI [36,30]. Furthermore, in cooperation with the research field of Explainable AI, attempts are being made to identify what knowledge has been learned by an AI [18,70] in order to be able to explain decisions made by an AI to, e.g., engineers or vehicle drivers [71].

Our developed relevance testing procedure and systematic method for generating reasonable abstract knowledge scenarios shall support these challenges. By identifying whether an AI applies the infused knowledge and, at the same time, satisfies system requirements, we support the verification and validation of knowledge-infused AI. Furthermore, with the identification of relevant knowledge, we provide a basis for designing and verifying explanations in the research field of Explainable AI.

8.2. Scenario specification

Two paths for the specification of traffic scenarios have been established in the literature. The first is the textual specification. An established standard is OpenSCENARIO [51] from the Association for Standardization of Automation and Measuring Systems (ASAM). This standard can be used to specify concrete scenarios with specific maneuvers. Well-known simulation platforms such as CARLA [50] and IPG Carmaker [72] already offer robust APIs for simulating OpenSCENARIO specifications. The Scenic framework [73], which focuses on the derivation of concrete and simulatable test scenarios, enables the definition of maneuvers and abstract scenarios using a probabilistic programming language. The simulation platform CARLA [50] offers a Python API, which enables the specification of concrete, directly simulatable scenarios.

In contrast to the established textual specification languages, our developed method for generating abstract test scenarios uses the visual yet formal specification language Traffic Sequence Charts (TSCs) [24]. As motivated in Section 5, we see a visual specification of traffic scenarios as an advantage when working with non-computer scientists. Especially in the shift of responsibility for essential driving decisions from humans to AI driving functions, it will be necessary to involve interdisciplinary experts to infuse social and societal norms and psychological or physiological aspects into an AI. We believe that a visual specification can help with interdisciplinary communication and thus avoid specification errors, facilitate final cross-checks of a specification, and make the extension of a specification more efficient.

Given a formal specification and a map, Klischat et al. in [74] visually specify traffic scenarios using Lanelets. Logical predicates are converted into mixed-integer logic, and a quadratic optimization problem is formed, the solutions of which are concrete scenarios. In contrast, we use Traffic Sequence Charts to specify formal specifications directly linked to a traffic scenario, and we do not have to commit to a specific map. Multi-Lane Spatial Logic (MLSL) [75] is a further visual specification language. The language offers the possibility to specify system requirements for system controllers in cooperative systems on urban road intersections. Unlike TSCs, which focus on spatio-temporal properties between road users over time, MLSL so far solely is able to reason about movement authority. Goyal et al. specify abstract scenarios in [76] using a 3x3 grid editor. The developed VIVAS framework thus enables the specification of LTL-based abstract scenarios focusing on discrete-time system requirements. Considering the possible upcoming interdisciplinary requirement elicitation of an AI driving function for essential driving decisions, system requirements are specified intuitively in continuous time. Using TSCs, continuous-time system requirements [25] can be specified in abstract scenarios and consequently discretized for concrete test scenarios. In addition, the TSC language also offers the option of discrete-time specifications.

8.3. Scenario-based testing and monitoring

Scenario-based testing is an established methodology [77]. In combination with simulation, the method offers a robust, cost- and time-efficient solution for developing highly automated and autonomous driving functions for strategic testing to verify compliance with system requirements [29]. With the development of AI driving functions, the black box property, and huge input space, formal verification methods are often inefficient or not applicable at all. Despite this, to check the system behavior for compliance with system requirements, the field provides solutions for runtime monitoring, which checks the behavior of a driving function at runtime in a simulated test scenario. There are two approaches here: Declarative temporal languages such as linear temporal logic (LTL) and signal temporal logic (STL). They offer the possibility to formalize system requirements with temporal properties, and there already

exists a wide range of monitoring applications [78–80]. Extensions for the specification of spatial properties, e.g., STSL [81] and SpaTeL [82], and SSSL [83] have also been developed. The synthesis of runtime monitors based on these languages already exists for various areas such as program verification and the verification of cyber-physical systems [84]. Zapridou et al. present in [85] a runtime monitoring based on STL-based system requirements for an Adaptive Cruise Control of an autonomous vehicle. Their goal was to monitor the robustness of PID controller requirements (invariant over speed and distance) inside the vehicle. Another example is the framework called BARK, presented in [86], which synthesizes monitors for LTL-based traffic rules in order to monitor traffic behavior at runtime.

The second approach is based on executable languages [84] such as automata. A wide range of solutions also exists in this area [3,87]. In addition, many translation schemes are based on declaratively specified system requirements [88,89]. Recently, Goyal et al. presented a conceptual runtime monitoring for abstract test scenarios based on LTL-based system requirements [76] and synthetization to state machines using the NuRV framework [90]. Since we use the visual yet formal specification language TSC for specification of system requirements and knowledge in abstract scenarios, we also use the existing TSC runtime monitoring [32] (new developments are submitted in the same Special Issue as this work). With our relevance testing procedure utilizing the combination of scenario-based testing and runtime monitoring, we show a direction in which the research field can develop solutions regarding the trustworthiness and system acceptance of knowledge-infused AI driving functions.

9. Conclusion and future work

This work is concerned with assessing the relevance of prior knowledge infused into an AI driving function to enable requirement satisfaction. For the first time, established notions of relevance in the field of Information Retrieval (IR) are related to development phases of knowledge-infused AI driving functions, and a suitable notion of *relevant knowledge* will be derived. A procedure for scenario-based testing is presented. This procedure checks knowledge conformance and requirement satisfaction of knowledge-infused AI driving functions. Based on this, we provide statements about the validity of previous development phases and indicate the relevance of infused knowledge of an AI driving function. Additionally, we present a systematic method for generating abstract knowledge scenarios based on Traffic Sequence Charts (TSCs). They also enable an efficient application of our relevance testing procedure. Finally, we presented a case study of the systematic derivation of abstract knowledge scenarios in combination with the presented relevance testing procedure. We demonstrated the feasibility of combining our presented methods and the unique value of distinguishing causal relations that are relevant for satisfying our example requirements from statistical correlations. These methods provide a formal combination of requirements and identified knowledge, as well as a testing procedure addressing the current challenges of ML practitioners.

The relevance testing procedure can be used in scenario-based development of knowledge-infused AI that uses prior domain knowledge to satisfy its requirements. This procedure can currently only be used for comparisons of specified behavior sets. Furthermore, the systematic derivation of abstract knowledge scenarios can be used if both a requirement specification and a knowledge specification are available as TSCs. Currently, the systematic derivation has been developed solely for the TSC language.

With this work, we provide a valuable contribution to the successful application of knowledge-infused AI driving functions in public transportation. Our notion of *relevant knowledge* for knowledge-infused AI, the relevance testing procedure, and the systematic method for generating abstract knowledge scenarios are able to address current challenges in the development and analysis of such knowledge-infused AI driving functions. Identifying relevant knowledge provides a foundation for knowledge bases that can be used for future AI developments and for the design and verification of explanations in the research field of Explainable AI.

In the future, the usefulness and integration of the presented relevance testing procedure can be evaluated using the presented toolchain in a larger case study developing a complex AI driving function. In addition, our relevance testing procedure could also be extended to handle metrics measuring how robust a requirement is satisfied. Consequently, relevant knowledge could be identified that improves existing AI driving functions to satisfy requirements more robustly. For example, permanently driving on the lane but close to the lane separator is less acceptable compared to driving mostly in the middle of the lane, although both would satisfy the requirement to stay on the lane. Given relevant knowledge, it can be investigated which different representations and information (i.e., used dataset, architecture, test suite with coverage information, etc.) a knowledge base needs to be widely applicable. Furthermore, it can be investigated how relevant knowledge can be used for the explanation of AI behavior. In addition, a method for validating explanations based on relevant knowledge can be developed.

CRedit authorship contribution statement

Dominik Grundt: Writing – original draft, Visualization, Validation, Methodology, Investigation, Conceptualization. **Astrid Rakow:** Writing – review & editing, Methodology, Formal analysis, Conceptualization. **Philipp Borchers:** Data curation. **Eike Möhlmann:** Writing – review & editing, Validation, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

The research leading to these results is funded by the German Federal Ministry for Economic Affairs and Climate Action within the project “NXT GEN AI METHODS – Generative Methoden für Perzeption, Prädiktion und Planung” (grant agreement No. 19A23014G).

References

- [1] Y.K. Dwivedi, A. Sharma, N.P. Rana, M. Giannakis, P. Goel, V. Dutot, Evolution of artificial intelligence research in technological forecasting and social change: research topics, trends, and future directions, *Technol. Forecast. Soc. Change* 192 (2023) 122579, <https://doi.org/10.1016/j.techfore.2023.122579>.
- [2] H. Elshohly, A. Azar, A. Shahin, O. Elsharkawy, H. Hassan, Path Planning of a Self Driving Vehicle Using Artificial Intelligence Techniques and Machine Vision, 2020, pp. 532–542, https://doi.org/10.1007/978-3-030-44289-7_50.
- [3] A. Gupta, A. Anpalagan, L. Guan, A.S. Khwaja, Deep learning for object detection and scene perception in self-driving cars: survey, challenges, and open issues, *Array* 10 (2021), <https://doi.org/10.1016/j.array.2021.100057>.
- [4] W. Schwarting, J. Alonso-Mora, D. Rus, Planning and decision-making for autonomous vehicles, *Ann. Rev. Control Robot. Auton. Syst.* 1 (2018) 187–210, <https://doi.org/10.1146/annurev-control-060117-105157>.
- [5] Y. Ma, Z. Wang, H. Yang, L. Yang, Artificial intelligence applications in the development of autonomous vehicles: a survey, *IEEE/CAA J. Autom. Sin.* 7 (2020) 315–329, <https://doi.org/10.1109/JAS.2020.1003021>.
- [6] C.M. de Melo, A. Torralba, L. Guibas, J. DiCarlo, R. Chellappa, J. Hodgins, Next-generation deep learning based on simulators and synthetic data, *Trends Cogn. Sci.* 26 (2) (2022) 174–187, <https://doi.org/10.1016/j.tics.2021.11.008>.
- [7] K.I. Wissen, Automotive AI powered by knowledge, <https://www.kiwissen.de>, 2024.
- [8] M. Gaur, A. Sheth, Building trustworthy NeuroSymbolic AI systems: consistency, reliability, explainability, and safety, *AI Mag.* 45 (1) (2024) 139–155, <https://doi.org/10.1002/aaai.12149>.
- [9] P.M. Fernandes, F.C. Santos, M. Lopes, Norms for beneficial A.I.: a computational analysis of the societal value alignment problem, *AI Commun.* 33 (3–6) (2020) 155–171, <https://doi.org/10.3233/AIC-201502>.
- [10] L. von Rueden, S. Mayer, K. Beckh, B. Georgiev, S. Giesselbach, R. Heese, B. Kirsch, J. Pfrommer, A. Pick, R. Ramamurthy, M. Walczak, J. Garcke, C. Bauckhage, J. Schuecker, Informed machine learning – a taxonomy and survey of integrating prior knowledge into learning systems, *IEEE Trans. Knowl. Data Eng.* 35 (1) (2023) 614–633, <https://doi.org/10.1109/TKDE.2021.3079836>.
- [11] A. Borghesi, F. Baldo, M. Milano, Improving deep learning models via constraint-based domain knowledge: a brief survey, arXiv:2005.10691 [abs], <https://api.semanticscholar.org/CorpusID:218763650>, 2020.
- [12] E. de Bézenac, A. Pajot, P. Gallinari, Deep learning for physical processes: incorporating prior scientific knowledge, *J. Stat. Mech. Theory Exp.* 2019 (12) (2019) 124009, <https://doi.org/10.1088/1742-5468/ab3195>.
- [13] S.L. Jurj, D. Grundt, T. Werner, P. Borchers, K. Rothemann, E. Möhlmann, Increasing the safety of adaptive cruise control using physics-guided reinforcement learning, *Energies* 14 (22) (2021), <https://doi.org/10.3390/en14227572>.
- [14] P. Borchers, W. Hagemann, D. Grundt, T. Werner, J. Müller, Using traffic sequence charts for knowledge formalization and AI-application, in: K. Arai (Ed.), *Intelligent Systems and Applications*, Springer Nature, Switzerland, Cham, 2024, pp. 198–220, https://doi.org/10.1007/978-3-031-66428-1_12.
- [15] B. Yet, Z. Perkins, N. Fenton, N. Tai, W. Marsh, Not just data: a method for improving prediction with knowledge, *J. Biomed. Inform.* 48 (2014) 28–37, <https://doi.org/10.1016/j.jbi.2013.10.012>.
- [16] M.v. Kurnatowski, J. Schmid, P. Link, R. Zache, L. Morand, T. Kraft, I. Schmidt, J. Schwientek, A. Stoll, Compensating data shortages in manufacturing with monotonicity knowledge, *Algorithms* 14 (12) (2021), <https://doi.org/10.3390/a14120345>.
- [17] R. Kaplan, C. Sauer, A. Sosa, Beating Atari with natural language guided reinforcement learning, arXiv:1704.05539 [abs], <https://api.semanticscholar.org/CorpusID:6022828>, 2017.
- [18] J. Wörmann, et al., Knowledge augmented machine learning with applications in autonomous driving: a survey, arXiv:2205.04712, 2023.
- [19] H.-M. Heyn, E. Knauss, I. Mallewaran, S. Dinakaran, An investigation of challenges encountered when specifying training data and runtime monitors for safety critical ml applications, in: A. Ferrari, B. Penzenstadler (Eds.), *Requirements Engineering: Foundation for Software Quality*, Springer Nature Switzerland, Cham, 2023.
- [20] A. Barredo Arrieta, N. Díaz-Rodríguez, J. Del Ser, A. Bennetot, S. Tabik, A. Barbado, S. Garcia, S. Gil-Lopez, D. Molina, R. Benjamins, R. Chatila, F. Herrera, Explainable artificial intelligence (XAI): concepts, taxonomies, opportunities and challenges toward responsible AI, *Inf. Fusion* 58 (2020) 82–115, <https://doi.org/10.1016/j.inffus.2019.12.012>.
- [21] M. Schwammberger, V. Klös, From specification models to explanation models: an extraction and refinement process for timed automata, in: M. Luckcuck, M. Farrell (Eds.), *Proceedings Fourth International Workshop on Formal Methods for Autonomous Systems (FMAS) and Fourth International Workshop on Automated and verifiable Software sYstem DEvelopment (ASYDE)*, FMAS/ASYDE@SEFM 2022, and Fourth International Workshop on Automated and verifiable Software sYstem DEvelopment (ASYDE), Berlin, Germany, 26th and 27th of September 2022, in: EPTCS, vol. 371, 2022, pp. 20–37, <https://doi.org/10.4204/EPTCS.371.2>.
- [22] ISO13586:2000(E), *Scenario-Based Safety Evaluation Framework for Automated Driving Systems*, Standard, International Organization for Standardization, Geneva, CH, Mar. 2023.
- [23] C. Neurohr, L. Westhofen, T. Henning, T. de Graaff, E. Möhlmann, E. Böde, Fundamental considerations around scenario-based testing for automated driving, in: 2020 IEEE Intelligent Vehicles Symposium (IV), 2020, pp. 121–127, <https://doi.org/10.1109/IV47402.2020.9304823>.
- [24] W. Damm, S. Kemper, E. Möhlmann, T. Peikenkamp, A. Rakow, Traffic Sequence Charts - from Visualization to Semantics, 2017, <https://doi.org/10.13140/RG.2.2.15190.42563>.
- [25] J.S. Becker, Partial consistency for requirement engineering with traffic sequence charts, in: Combined Proceedings of the Workshops at Software Engineering 2020 Co-located with the German Software Engineering Conference 2020 (SE 2020), Innsbruck, Österreich, in: CEUR Workshop Proceedings, vol. 2581, March 05, 2020, <http://ceur-ws.org/Vol-2581/ase2020paper1.pdf>, 2020.
- [26] G.G. Towell, J.W. Shavlik, Knowledge-based artificial neural networks, *Artif. Intell.* 70 (1) (1994) 119–165, [https://doi.org/10.1016/0004-3702\(94\)90105-8](https://doi.org/10.1016/0004-3702(94)90105-8).
- [27] P. Hamet, J. Tremblay, Artificial intelligence in medicine, in: *Insights into the Future of Medicine: Technologies, Concepts, and Integration*, *Metabolism* 69 (2017) S36–S40, <https://doi.org/10.1016/j.metabol.2017.01.011>.
- [28] J. Perez-Cerrolaza, J. Abella, M. Borg, C. Donzella, J. Cerquides, F.J. Cazorla, C. Englund, M. Tauber, G. Nikolakopoulos, J.L. Flores, Artificial intelligence for safety-critical systems in industrial and transportation domains: a survey, *ACM Comput. Surv.* 56 (7) (Apr. 2024), <https://doi.org/10.1145/3626314>.
- [29] C. Neurohr, L. Westhofen, M. Butz, M. Bollmann, U. Eberle, R. Galbas, Criticality analysis for the verification and validation of automated vehicles, *IEEE Access* (2021) 1, <https://doi.org/10.1109/ACCESS.2021.3053159>.
- [30] K. Manas, A. Paschke, Semantic role assisted natural language rule formalization for intelligent vehicle, in: A. Fensel, A. Ozaki, D. Roman, A. Soylu (Eds.), *Rules and Reasoning*, Springer Nature, Switzerland, Cham, 2023, pp. 175–189, https://doi.org/10.1007/978-3-031-45072-3_13.
- [31] C. Szegedy, A promising path towards autoformalization and general artificial intelligence, in: C. Benz Müller, B. Miller (Eds.), *Intelligent Computer Mathematics*, Springer International Publishing, Cham, 2020, pp. 3–20, https://doi.org/10.1007/978-3-030-53518-6_1.

- [32] D. Grundt, A. Köhne, I. Saxena, R. Stemmer, B. Westphal, E. Möhlmann, Towards runtime monitoring of complex system requirements for autonomous driving functions, in: M. Luckcuck, M. Farrell (Eds.), Proceedings Fourth International Workshop on Formal Methods for Autonomous Systems (FMAS) and Fourth International Workshop on Automated and verifiable Software sYstem DEvelopment (ASYDE), Berlin, Germany, 26th and 27th of September 2022, in: Electronic Proceedings in Theoretical Computer Science, vol. 371, Open Publishing Association, 2022, pp. 53–61, <https://doi.org/10.4204/EPTCS.371.4>.
- [33] L. Westhofen, I. Stierand, J.S. Becker, E. Möhlmann, W. Hagemann, Towards a congruent interpretation of traffic rules for automated driving - experiences and challenges, in: G. Borges, K. Satoh, E. Schweighofer (Eds.), Proceedings of the International Workshop on Methodologies for Translating Legal Norms into Formal Representations (LN2FR 2022) in Association with the 35th International Conference on Legal Knowledge and Information Systems (JURIX 2022), 2022, pp. 8–21, <https://elib.dlr.de/193009>.
- [34] G. Borges, K. Satoh, E. Schweighofer, Proceedings of the international workshop on methodologies for translating legal norms into formal representations (LN2FR 2022) in association with 35th international conference on legal knowledge and information systems (JURIX 2022), CoRR, arXiv:abs/2305.12203, arXiv:2305.12203, <https://doi.org/10.48550/ARXIV.2305.12203>, 2023.
- [35] A.H. Khan, M. Munir, L. van Elst, A. Dengel, F2DNet: fast focal detection network for pedestrian detection, in: 2022 26th International Conference on Pattern Recognition (ICPR), 2022, pp. 4658–4664, <https://doi.org/10.1109/ICPR56361.2022.9956732>.
- [36] H. Agarwal, C. Brunner, T. Latka, S. Rudolph, A causal model for physics-conform vehicle trajectories, in: 2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC), 2023, pp. 4980–4987, <https://doi.org/10.1109/ITSC57777.2023.10422314>.
- [37] J. Ouaknine, J. Worrell, Some recent results in metric temporal logic, in: F. Cassez, C. Jard (Eds.), Formal Modeling and Analysis of Timed Systems, Springer, Berlin Heidelberg, Berlin, Heidelberg, 2008, pp. 1–13, https://doi.org/10.1007/978-3-540-85778-5_1.
- [38] A.M. Rees, The relevance of relevance to the testing and evaluation of document retrieval systems, in: Aslib Proceedings, Vol. 18-11, 1966, pp. 316–324, <https://doi.org/10.1108/eb050068>.
- [39] G.J. Baumanis, A.M. Rees, D.G. Schultz, A field experimental approach to the study of relevance assessments in relation to document searching final report to the national science foundation, <https://api.semanticscholar.org/CorpusID:61043298>, 1967.
- [40] T. Saracevic, RELEVANCE: a review of and a framework for the thinking on the notion in information science, J. Am. Soc. Inf. Sci. 26 (1975) 321–343, <https://doi.org/10.1002/asi.4630260604>.
- [41] L. Schamber, M.B. Eisenberg, M.S. Nilan, A re-examination of relevance: toward a dynamic, situational definition, Inf. Process. Manag. 26 (6) (1990) 755–776, [https://doi.org/10.1016/0306-4573\(90\)90050-C](https://doi.org/10.1016/0306-4573(90)90050-C).
- [42] T. Saracevic, Relevance: a review of the literature and a framework for thinking on the notion in information science, J. Am. Soc. Inf. Sci. Technol. 58 (2007) 2126, <https://doi.org/10.1002/asi.20681>.
- [43] P. Ingwersen, K. Järvelin, The Turn: integration of information seeking and retrieval, in: Context, in: The Information Retrieval Series, Springer-Verlag, Berlin, Heidelberg, 2005, <https://doi.org/10.1007/1-4020-3851-8>.
- [44] P. Wilson, Situational relevance, Inf. Storage Retr. 9 (8) (1973) 457–471, [https://doi.org/10.1016/0020-0271\(73\)90096-X](https://doi.org/10.1016/0020-0271(73)90096-X).
- [45] M. Czaronis, M. Kritzman, D. Turkington, Relevance-based prediction: a transparent and adaptive alternative to machine learning, J. Finance Data Sci. 5 (12 2022), <https://doi.org/10.3905/jfds.2022.1.110>.
- [46] P. Borchers, T. Koopmann, L. Westhofen, J.S. Becker, L. Putze, D. Grundt, T. de Graaff, V. Kalwa, C. Neurohr, TSC2CARLA: an abstract scenario-based verification toolchain for automated driving systems, Sci. Comput. Program. 242 (2025) 103256, <https://doi.org/10.1016/j.scico.2024.103256>.
- [47] A. Rakow, A notion of relevance for safety critical autonomous systems, in: M. Fränzle, J. Niehaus, B. Westphal (Eds.), Engineering Safe and Trustworthy Cyber Physical Systems – Essays Dedicated to Werner Damm on the Occasion of His 71st Birthday, Springer Nature Switzerland AG, 2024, Accepted for publication.
- [48] J. Becker, T. Koopmann, B. Neurohr, C. Neurohr, L. Westhofen, B. Wirtz, E. Böde, W. Damm, Simulation of Abstract Scenarios: Towards Automated Tooling in Criticality Analysis, 2022, pp. 42–51, <https://doi.org/10.5281/zenodo.5907154>.
- [49] Association for Standardization of Automation and Measuring Systems (ASAM), ASAM OpenSCENARIO V1.3, <https://www.asam.net/standards/detail/openscenario-xml/>, 2024. (Accessed 13 June 2024).
- [50] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, V. Koltun, CARLA: an open urban driving simulator, in: Proceedings of the 1st Annual Conference on Robot Learning, 2017, pp. 1–16, <https://doi.org/10.48550/ARXIV.1711.03938>.
- [51] Association for Standardization of Automation and Measuring Systems (ASAM), ASAM OpenSCENARIO V2.0, <https://www.asam.net/project-detail/asam-openscenario-v20-1/>, 2022. (Accessed 14 May 2024).
- [52] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, N. Dormann, Stable-baselines3: reliable reinforcement learning implementations, J. Mach. Learn. Res. 22 (1) (Jan 2021), <https://doi.org/10.5555/3546258.3546526>.
- [53] OpenAI Gymnasium, Framework documentation, <https://gymnasium.farama.org/>, 2023.
- [54] Mario Theers, Mankaran Singh, Kinematic bicycle model, <https://thomasfermi.github.io/Algorithms-for-Automated-Driving/Control/BicycleModel.html>, 2020.
- [55] S. Ding, M. Abdel-Aty, N. Barbour, D. Wang, Z. Wang, O. Zheng, Exploratory analysis of injury severity under different levels of driving automation (sae levels 2 and 4) using multi-source data, Accid. Anal. Prev. 206 (2024) 107692, <https://doi.org/10.1016/j.aap.2024.107692>.
- [56] S. Babisch, C. Neurohr, L. Westhofen, S. Schoenawa, H. Liers, Leveraging the gidas database for the criticality analysis of automated driving systems, J. Adv. Transp. (May 2023).
- [57] Y. Song, M.V. Chitturi, D.A. Noyce, Automated vehicle crash sequences: patterns and potential uses in safety testing, Accid. Anal. Prev. 153 (2021) 106017, <https://doi.org/10.1016/j.aap.2021.106017>.
- [58] B. Kramer, C. Neurohr, M. Büker, E. Böde, M. Fränzle, W. Damm, Identification and quantification of hazardous scenarios for automated driving, in: Model-Based Safety and Assessment: 7th International Symposium, IMBSA 2020, Lisbon, Portugal, September 14–16, 2020, Proceedings, Springer-Verlag, Berlin, Heidelberg, 2020, pp. 163–178, https://doi.org/10.1007/978-3-030-58920-2_11.
- [59] Q. Xue, X. Ouyang, Y. Zhao, W. Guo, Effect of situation kinematics on drivers' rear-end collision avoidance behaviour—a combined effect of visual looming, speed, and distance analysis, Sustainability 14 (22) (2022), <https://doi.org/10.3390/su142215103>.
- [60] J. Li, F. Guo, Y. Zhou, W. Yang, D. Ni, Predicting the severity of traffic accidents on mountain freeways with dynamic traffic and weather data, Transp. Saf. Environ. 5 (4) (2023) tdad001, <https://doi.org/10.1093/tse/tdad001>.
- [61] N. Angelopoulos, J. Cussens, Bayesian learning of Bayesian networks with informative priors, Ann. Math. Artif. Intell. 54 (2008) 53–98, <https://doi.org/10.1007/s10472-009-9133-x>.
- [62] F. Lauer, G. Bloch, Incorporating prior knowledge in support vector machines for classification: a review, in: Progress in Modeling, Theory, and Application of Computational Intelligence, Neurocomputing 71 (7) (2008) 1578–1594, <https://doi.org/10.1016/j.neucom.2007.04.010>.
- [63] O.L. Mangasarian, E.W. Wild, Nonlinear knowledge-based classification, IEEE Trans. Neural Netw. 19 (10) (2008) 1826–1832, <https://doi.org/10.1109/TNN.2008.2005188>.
- [64] S. Reich, C. Cotter, Probabilistic Forecasting and Bayesian Data Assimilation, Cambridge University Press, 2015, <https://doi.org/10.1017/CBO9781107706804>.
- [65] X. Zhang, J. Zhang, K. Sun, X. Yang, C. Dai, Y. Guo, Integrated multi-omics analysis using variational autoencoders: application to pan-cancer classification, in: 2019 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), 2019, pp. 765–769, <https://doi.org/10.1109/BIBM47256.2019.8983228>.
- [66] S. Dieleman, J.D. Fauw, K. Kavukcuoglu, Exploiting cyclic symmetry in convolutional neural networks, in: M.F. Balcan, K.Q. Weinberger (Eds.), Proceedings of the 33rd International Conference on Machine Learning, in: Proceedings of Machine Learning Research, vol. 48, PMLR, New York, New York, USA, 2016, pp. 1889–1898, <http://proceedings.mlr.press/v48/dieleman16.pdf>, Corpus ID: 8569309.
- [67] L. Yang, D. Zhang, G.E. Karniadakis, Physics-informed generative adversarial networks for stochastic differential equations, SIAM J. Sci. Comput. 42 (1) (2020) A292–A317, <https://doi.org/10.1137/18M1225409>.

- [68] D. Bogdoll, J. Qin, M. Nekolla, A. Abouelazm, T. Joseph, J.M. Zöllner, Informed reinforcement learning for situation-aware traffic rule exceptions, arXiv: 2402.04168 [abs], <https://api.semanticscholar.org/CorpusID:267499566>, 2024.
- [69] J. Collenette, L.A. Dennis, M. Fisher, Advising autonomous cars about the rules of the road, in: M. Luckcuck, M. Farrell (Eds.), Proceedings Fourth International Workshop on Formal Methods for Autonomous Systems (FMAS) and Fourth International Workshop on Automated and Verifiable Software sYstem DEvelopment (ASYDE), Berlin, Germany, 26th and 27th of September 2022, in: Electronic Proceedings in Theoretical Computer Science, vol. 371, Open Publishing Association, 2022, pp. 62–76, <https://doi.org/10.4204/EPTCS.371.5>.
- [70] G. Mikriukov, G. Schwalbe, C. Hellert, K. Bade, Evaluating the Stability of Semantic Concept Representations in CNNs for Robust Explainability, Springer Nature, Switzerland, 2023, pp. 499–524, https://doi.org/10.1007/978-3-031-44067-0_26.
- [71] M. Gaur, U. Kursuncu, A. Sheth, R. Wickramarachchi, S. Yadav, Knowledge-infused deep learning, in: Proceedings of the 31st ACM Conference on Hypertext and Social Media, HT '20, Association for Computing Machinery, New York, NY, USA, 2020, pp. 309–310, <https://doi.org/10.1145/3372923.3404862>.
- [72] IPG Automotive, IPG CarMaker, <https://www.ipg-automotive.com/en/products-solutions/software/carmaker/>, last visited: 06/2024.
- [73] D.J. Fremont, T. Drossi, S. Ghosh, X. Yue, A.L. Sangiovanni-Vincentelli, S.A. Seshia, Scenic: a language for scenario specification and scene generation, in: Proceedings of the 40th ACM SIGPLAN Conference on Programming Language Design and Implementation, PLDI 2019, Association for Computing Machinery, New York, NY, USA, 2019, pp. 63–78, <https://doi.org/10.1145/3314221.3314633>.
- [74] M. Klischat, M. Althoff, Synthesizing traffic scenarios from formal specifications for testing automated vehicles, in: 2020 IEEE Intelligent Vehicles Symposium (IV), 2020, pp. 2065–2072, <https://doi.org/10.1109/IV47402.2020.9304617>.
- [75] M. Schwammbberger, Distributed Controllers for Provably Safe, Live and Fair Autonomous Car Manoeuvres in Urban Traffic, Ph.D. thesis, Oldenburg University, Germany, 2021, <https://oops.uni-oldenburg.de/4961/>.
- [76] S. Goyal, A. Griggio, J. Kimblad, S. Tonetta, Automatic generation of scenarios for system-level simulation-based verification of autonomous driving systems, in: M. Farrell, M. Luckcuck, M. Gleirscher, M. Schwammbberger (Eds.), Proceedings Fifth International Workshop on Formal Methods for Autonomous Systems, Leiden, the Netherlands, 15th and 16th of November 2023, in: Electronic Proceedings in Theoretical Computer Science, vol. 395, Open Publishing Association, 2023, pp. 113–129, <https://doi.org/10.4204/EPTCS.395.8>.
- [77] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, F. Diermeyer, Survey on scenario-based safety assessment of automated vehicles, IEEE Access 8 (2020) 87456–87477, <https://doi.org/10.1109/ACCESS.2020.2993730>.
- [78] A. Donzé, O. Maler, Robust satisfaction of temporal logic over real-valued signals, in: International Conference on Formal Modeling and Analysis of Timed Systems, Springer, 2010, pp. 92–106, https://doi.org/10.1007/978-3-642-15297-9_9.
- [79] A. Donzé, T. Ferrère, O. Maler, Efficient robust monitoring for STL, in: Computer Aided Verification, Springer International Publishing, 2013, pp. 264–279, https://doi.org/10.1007/978-3-642-39799-8_19.
- [80] J.V. Deshmukh, A. Donzé, S. Ghosh, X. Jin, G. Juniwal, S.A. Seshia, Robust online monitoring of signal temporal logic, Form. Methods Syst. Des. (2017) 5–30, <https://doi.org/10.1007/s10703-017-0286-7>.
- [81] T. Li, J. Liu, J. Kang, H. Sun, W. Yin, X. Chen, H. Wang, STSL: a novel spatio-temporal specification language for cyber-physical systems, in: 2020 IEEE 20th International Conference on Software Quality, Reliability and Security (QRS), 2020, pp. 309–319, <https://doi.org/10.1109/QRS51102.2020.00048>.
- [82] I. Haghghi, A. Jones, Z. Kong, E. Bartocci, R. Gros, C. Belta, SpaTeL: a novel spatial-temporal logic and its applications to networked systems, in: Proceedings of the 18th International Conference on Hybrid Systems: Computation and Control, HSCC '15, Association for Computing Machinery, 2015, pp. 189–198, <https://doi.org/10.1145/2728606.2728633>.
- [83] L. Nenzi, L. Bortolussi, V. Ciancia, M. Loreti, M. Massink, Qualitative and quantitative monitoring of spatio-temporal properties with SSTL, Log. Methods Comput. Sci. 14 (2017), [https://doi.org/10.23638/LMCS-14\(4:2\)2018](https://doi.org/10.23638/LMCS-14(4:2)2018).
- [84] E. Bartocci, J.V. Deshmukh, A. Donzé, G. Fainekos, O. Maler, D. Ničković, S. Sankaranarayanan, Specification-based monitoring of cyber-physical systems: a survey on theory, tools and applications, in: Lectures on Runtime Verification, 2018, https://doi.org/10.1007/978-3-319-75632-5_5.
- [85] E. Zapridou, E. Bartocci, P. Katsaros, Runtime verification of autonomous driving systems in CARLA, in: J. Deshmukh, D. Ničković (Eds.), Runtime Verification, Springer International Publishing, Cham, 2020, pp. 172–183, https://doi.org/10.1007/978-3-030-60508-7_9.
- [86] K. Esterle, L. Gressenbuch, A. Knoll, Modeling and testing multi-agent traffic rules within interactive behavior planning, arXiv:2009.14186, 2020.
- [87] P. Zhang, W. Li, D. Wan, L. Grunske, Monitoring of probabilistic timed property sequence charts, Softw. Pract. Exp. 41 (7) (2011) 841–866, <https://doi.org/10.1002/spe.1038>.
- [88] T. Ferrère, O. Maler, D. Ničković, A. Pnueli, From real-time logic to timed automata, J. ACM 66 (2019), <https://doi.org/10.1145/3286976>.
- [89] A. Bauer, M. Leucker, C. Schallhart, Runtime verification for LTL and TLTL, ACM Trans. Softw. Eng. Methodol. 20 (2011), <https://doi.org/10.1145/2000799.2000800>.
- [90] A. Cimatti, C. Tian, S. Tonetta, Nurv: a nuxmv extension for runtime verification, in: B. Finkbeiner, L. Mariani (Eds.), Runtime Verification, Springer International Publishing, Cham, 2019, pp. 382–392, https://doi.org/10.1007/978-3-030-32079-9_23.