Context-aware, Ante-hoc Explanations of Driving Behaviour

Dominik Grundt Ishan Saxena Malte Petersen Bernd Westphal Eike Möhlmann
<firstname.lastname>@dlr.de

German Aerospace Center
Institute of Systems Engineering for Future Mobility
Oldenburg, Germany

Autonomous vehicles (*AVs*) must be both safe and trustworthy to gain social acceptance and become a viable option for everyday public transportation. Explanations about the system behaviour can increase safety and trust in *AVs*. Unfortunately, explaining the system behaviour of AI-based driving functions is particularly challenging, as decision-making processes are often opaque. The field of *Explainability Engineering* tackles this challenge by developing *explanation models* at design time. These models are designed from system design artefacts and stakeholder needs to develop *correct* and *good* explanations. To support this field, we propose an approach that enables context-aware, ante-hoc explanations of (un)expectable driving manoeuvres at runtime. The visual yet formal language Traffic Sequence Charts is used to formalise *explanation contexts*, as well as corresponding (un)expectable driving manoeuvres. A dedicated runtime monitoring enables context-recognition and ante-hoc presentation of explanations at runtime. In combination, we aim to support the bridging of *correct* and *good* explanations. Our method is demonstrated in a simulated overtaking.

The research leading to these results is funded by the German Federal Ministry of Education and Research under grant agreement No 16MEE044 (EdgeAI-Trust) and by the Chips Joint Undertaking under grant agreement No 101139892 (EdgeAI-Trust).

1 Introduction

To deploy autonomous vehicles (*AVs*) in public road traffic and to gain societal acceptance, they must be safe and trustworthy. Achieving these properties in road traffic is difficult for two main reasons. First, the inherently dynamic and open-world nature of road traffic introduces uncertainty, as *AVs* must operate under constantly changing and only partially predictable conditions. Second, the black-box character of many AI-based solutions limits transparency and interpretability, making it difficult to assess their correctness, especially crucial in safety-critical situations. In combination, these challenges make the specification of a complete formal behaviour model of *AVs* infeasible.

On the regulatory level, the upcoming European AI Act [1] mandates transparency, human oversight, and traceability for high-risk AI systems, including *AVs*. Moreover, studies such as [26] and [3] reveal that social acceptance of *AVs* remains limited, mainly due to a lack of trust. Consequently, there is an urgent need for reliable explanation methods in autonomous driving, even in the absence of a complete formal behavioural model.

The field of Explainability Engineering addresses this challenge by developing methods and tools to systematically design explanation models from system descriptions, stakeholder needs, and requirements at design time. The goal is to build a foundation for developing explanations that are both *correct* and *good* [29]. Research in this field enhances trust, facilitates safety validation, and fulfils regulatory requirements by making AI decisions interpretable, traceable and understandable for different stakeholders. Empirical research provides guidance on which explanations are effective in increasing trust.

Matt Luckcuck, Maike Schwammberger & Mengwei Xu (Eds.):
Seventh International Workshop on Formal Methods
for Autonomous Systems (FMAS 2025)
EPTCS 436, 2025, pp. 114–135, doi:10.4204/EPTCS.436.11

Meta-studies indicate that while system performance is important, trust in autonomous systems is significantly enhanced by behavioural explanations [5]. Explaining *expectable* manoeuvres helps stakeholders build confidence in an *AV* 's reliable and plausible operation [5]. Conversely, explaining *unexpectable* manoeuvres is crucial for maintaining trust in safety-critical situations [27]. For this work, we refer to *(un)expectable* driving manoeuvres as behaviour that is *(not-)permitted* according to an *AV* 's explanation model. Rather than *(un)expected*, the term *(un)expectable* emphasises model-based admissibility instead of compliance with predefined requirements, while also capturing runtime uncertainties of the domain and AI-based driving functions.

Furthermore, explanations are most effective when provided *before* an *AV* executes a manoeuvre [17]. We refer to such explanations as *ante-hoc*, meaning that information about (un)expectable behaviour is available prior to execution. Prior work also shows that, to be effective, trustworthy, and meaningful, explanations of *AV* behaviour must consider the surrounding traffic and environmental conditions [12, 23, 25]. We refer to explanations that include such factors as *context-aware*.

While frameworks such as MAB-EX [11] describe how explainability can be achieved at runtime, and literature in Explainability Engineering defines properties and requirements for *good* and *correct* explanations [29], concrete methods to operationalise these concepts for *AVs* are still lacking. In this work, we present a method to specify *context-aware* explanations of (un)expectable driving manoeuvres and to provide such explanations *ante-hoc* at runtime.

We use Traffic Sequence Charts (TSC) [15] to formally specify the *explanation context* (or short *context*), i.e., the traffic situation in which an explanation should be provided. TSC allow the specification of spatio-temporal properties, particularly of traffic scenarios, in both visual and formal form. Using TSC for context specification is beneficial for stakeholder needs, such as comprehensibility and goodness. For example, psychologists can assess both properties by investigation of the visual part of TSC without needing to understand the formal semantics. Further, we use TSC runtime monitoring [32, 20] to recognise the specified explanation context at runtime and trigger the ante-hoc presentation of an explanation. Finally, we show how (un)expectable driving manoeuvres can be specified with TSC.

Therefore, we address the challenges of (i) formally specifying explanation contexts for *AVs*, (ii) enabling ante-hoc explanations at runtime, and (iii) enabling different forms of an explanation about (un)expectable driving manoeuvres.

Outline. The paper is organised as follows: Section 2 provides preliminaries for our work. Section 3 discusses related work. Section 4 presents the formalisation of an *explanation context* using TSC. Section 5 presents TSC runtime monitoring for context-recognition and ante-hoc explanations at runtime. Section 6 presents the visual yet formal specification of (un)expectable driving manoeuvres in a context-aware manner using TSC. Section 7 demonstrates our approach within a simulated overtaking. Section 8 contains the conclusion and discusses future work.

2 Preliminaries

In this section, we give a brief introduction to the field of Explainability Engineering, including a characterisation of *explanations* and describe how our approach aligns with the MAB-EX Framework [11].

Explainability Engineering In [24], explainability was characterised as a non-functional requirement. With its recent classification as a requirement also at the regulatory level, explainability has become an engineering task for AI-based systems such as *AVs*. Presenting pure information without context does not provide the necessary explainability that creates traceability, transparency and comprehensibility of system behaviour [25]. As shown in [30], stakeholder-specific explanation models for such complex

systems need to be created systematically at design time. Explainability Engineering (EE) directly addresses this need for transparency and traceability at both regulatory [1] and societal levels [26, 3] for high-risk AI systems.

While methods in eXplainable AI (XAI) primarily aim at generating explanations for trained, complex AI models, these explanations are typically tied to a specific model and often remain interpretable only to experts [31]. In contrast, EE pursues the broader goal of creating explanations for the behaviour of entire AI-based systems or systems of systems, such as *AVs*.

An explanation model *EM* may be defined as a behavioural model of the system that captures causal relationships between events and system reactions. This enables the identification of possible causes for behaviours that require explanation [11]. Such models provide a conceptual foundation for delivering explanations to diverse stakeholders, from engineers needing technical traceability to passengers seeking intuitive justifications and regulators requiring formal evidence of compliance. In [30], Schwammberger and Klös showed how to derive explanation models in the context of autonomous agents. In this paper, we assume the existence of an *EM* that contains relations between traffic situations and corresponding (un)expectable driving manoeuvres.

Explanations in Explainability Engineering In our work, we use the following characterisation of explanations given by Schwammberger et al. as an extension of Bohlender and Köhl (see [12]):

Characterisation 1 (Explanation [31])**.** *An explanation E is characterised by (i) explananda X, or "phenomena", of the system of interest, (ii) a context C, (iii) a stakeholder group G for E, (iv) the goal θ of E for a stakeholder group G, and (v) the means M for producing E.*

In our approach, the explanandum X of an explanation E is which driving manoeuvres of an *AV* are (un)expectable in the current context C, where the context represents the *AV* 's current traffic situation. Using TSC with dedicated runtime monitoring, both the context C and the context-aware, ante-hoc presentation of explanations E can be formalised in the same specification language. Thus, for the explanation means M, we leverage the visual yet formal form of TSC to specify driving manoeuvres.

This setup is deliberately modular. The TSC-based specification of context C and its runtime monitoring can be replaced by any other context-recognition method. The TSC-specified driving manoeuvres continue to serve as the explanation means M. Conversely, the TSC-based context and monitoring can trigger context-aware, ante-hoc presentation of any other explanations. Regarding our method, the goal θ is to increase trust in *AVs* across different stakeholder groups G. However, explanation *goodness* [29] needs to be investigated by future empirical studies.

MAB-EX Framework In EE, the *MAB-EX* framework [11] (depicted in Fig. 1) provides a theoretical foundation for creating self-explanatory capabilities of systems at runtime. Our approach can be aligned with the four phases ***M****onitoring*, ***A****nalyse*, ***B****uild*, and ***EX****plain*.

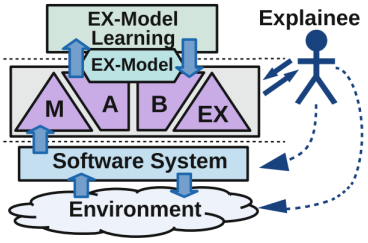


Figure 1: MAB-EX Framework [11] (concretised for software systems in [31]).

The system considered in this work is an *AV* interacting with its environment. The explanation model captures the causal factors underlying the *AV*'s behaviour. Within MAB-EX, a learning mechanism for the explanation model is foreseen, which can iteratively improve the *AV*'s explanation model for different stakeholders using operational data from the dynamic and uncertain automotive domain. Both the design of the explanation model and any model learning remain out of scope. In this work, we primarily operationalise the four runtime phases for context-aware, ante-hoc explanations of (un)expectable driving manoeuvres. The TSC runtime monitoring corresponds to the ***M****onitoring* phase, where surrounding traffic and the environment are

continuously observed. The subsequent context recognition, based on the TSC specification, maps to the ***A**nalyse* phase, where the observed data are checked for the phenomenon to be explained (the explanandum X). The TSC specification of traffic situations and corresponding (un)expectable manoeuvres, along with their runtime selection, belongs to the ***B**uild* phase, where explanations are developed from an explanation model. Finally, the visual or formal presentation of (un)expectable manoeuvres corresponds to the ***EX**plain* phase, communicating system behaviour in an interpretable and comprehensible way.

3 Related Work

Formal specification of spatio-temporal properties has already been used in the automotive domain to specify, e.g., system requirements based on languages such as LTL [28, 18] and STL [33, 4], as well as for complex system requirements with TSC [32, 20]. Since no language has emerged as standard, we employ TSC for context and manoeuvrer specification, whose visual and formal syntax may be able to address multiple stakeholder groups. Visual or formal XAI presentations of explanations, such as saliency maps, heatmaps, or decision-tree paths [2], target domain experts and require further interpretation. In contrast, TSC-based specifications allow for deciding in advance what to communicate, how, and to whom, tailoring the explanation to the need, context and goals. This aligns with Explainability Engineering principles of designing *correct* and *good* explanations [29].

Runtime Monitoring (RM) is a technique used to check whether a system's runtime behaviour complies with formally specified properties [8]. Particularly in safety-critical domains, RM ensures the correctness and safety of complex systems such as *AVs*. RM continuously observes system behaviour to detect deviations from specifications [9], complementing offline verification and allowing timely identification of unsafe states [22]. For this work, we use and adapt an already developed TSC runtime monitoring [20, 32] for recognising an explanation context specified using TSC.

Schwammberger et al. [30] demonstrate a possible derivation method for explanation models that considers different stakeholder groups and present a case study where a timed automaton models a crossing protocol for urban intersection turn manoeuvres. Their case study discusses the MAB-EX phases and explanation forms abstractly for multiple stakeholder groups. In a recent extension of the MAB-EX framework [31], the authors introduce the concept of an *explanation history* to support the recognition of recurring contexts and dynamically adapt explanations accordingly. This approach enables the generation of *global* explanations that generalise across a variety of similar contexts.

This work can be seen as the first concrete operationalisation of MAB-EX runtime phases to provide context-aware, ante-hoc explanations of (un)expectable *AV* manoeuvres at runtime. Complementing prior work on deriving explanation models for different stakeholder groups [30] and on extending the MAB-EX framework with an explanation history [31], our approach focuses on implementing the runtime phases in a context-aware manner. Together, these perspectives form a coherent research trajectory: stakeholder-specific explanation models provide the conceptual foundation, our operationalisation demonstrates runtime applicability, and extensions like explanation history enable adaptive refinement of explanations.

4 Traffic Sequence Charts for Context-Aware Explanations

In the following, we show how TSC can be used to formalise an explanation context to enable context-recognition at runtime and specify context-aware explanations.

Recall that we assume the existence of an explanation model EM that contains relations between

traffic situations and corresponding (un-)expectable driving manoeuvres, created by system and domain experts at design time. Considering the characterisation of an explanation (see Char. 1), we define the context C of an explanation E about the behaviour of an AV as the specification of a traffic situation. A traffic situation of an AV consists of the surrounding environment (e.g., lanes, markings, road signs) and traffic participants which interact with the environment. Traffic participants are constrained by behavioural and physical rules in the environment, and have attributes such as speed and position. Functional descriptions of traffic situations or a context C for our explanation E can be as follows:

Example 1 (Context)**.** *AV is driving towards a slow driving vehicle (slower than the speed limit) in its lane on a two-lane carriageway. There is no other vehicle on the adjacent left lane of AV. The distance between AV and the other vehicle is less than 25 metres.*

We refer to a concrete instance of context C, in which an explanation E of AV is to be presented, as a *concrete traffic situation*. A concrete traffic situation captures the spatial relations between traffic participants and the environment at a specific point in time, including the fixed state of object attributes.

To formalise the concrete traffic situation as context C of an explanation E, we model the traffic environment of AV, its traffic participants and their respective attributes in an object model $OM = (\mathcal{T}, \mathcal{C}, F, Pred)$, where $\mathcal{T}$ is a set of basic types, $\mathcal{C}$ is a set of object types (or classes), F is a set of typed function symbols, and $Pred$ is a set of typed predicate symbols. With $attr(\mathbf{C})$, we describe the finite set of typed attributes for each object class $\mathbf{C} \in \mathcal{C}$. Now, given an object model OM, we define a concrete traffic situation as a function $\sigma : ID \rightarrow attr(\mathbf{C}) \rightarrow \mathcal{D}$, where $\mathbf{ID}$ denotes the finite set of object identities (i.e., the traffic objects involved in the situation), $attr(\mathbf{C})$ the attributes of the context, and $\mathcal{D}$ the domain of valid, type-consistent values.

In the following, we briefly introduce TSC and show how they can be used to formalise an explanation context C as traffic situation specification. Such a specification characterises not a single, but a set of concrete traffic situations that an AV might encounter, as in our example context.

4.1 Traffic Sequence Charts

TSC is a visual and formal language for specifying spatio-temporal properties in traffic environments. The simplest chart, a *Basic Chart*, contains a single *invariant node*. Each node encapsulates a *Spatial View* (SV), the core visual formalism, expressing spatial relations (e.g., distances) and object-attribute predicates (e.g., velocity, acceleration), combined with Boolean operators.

Given the underlying time model $\mathbb{T}$ of TSC with continuous time semantics, any constraint holds for a non-empty time interval. Therefore, a Basic Chart, i.e. an invariant node containing an SV, specifies propositional constraints that hold invariantly over an interval $[b, e] \subset \mathbb{T}$ with $e > b$.

TSC can be used to formalise the context C as a traffic situation specification, i.e., a set of well-typed predicate logic formulae defined over an OM. Together with the aforementioned time model $\mathbb{T}$, TSC allow the specification of so-called abstract traffic situations S.

A so-called *symbol dictionary* is essential for the visual syntax of SV. Object symbols from the symbol dictionary can be placed on a rectangular canvas, which exists within the corresponding invariant node. Each object symbol is assigned to exactly one object type $\mathbf{C}$ from OM and may have unique logical object variables, often associated with an object's $id \in ID$. Anchors are always required and necessary for expressing spatial properties. Each object symbol has at least one anchor, e.g. in the centre of the symbol. Each anchor is linked to a position attribute $pos \in \mathbb{R}^2$ of an object type instance, which is always required. Exceptions are the so-called line anchors, which are fixed in one dimension on the 2-dimensional canvas. These can be used, for example, to logically specify the boundaries of a carriageway. Thus, by placing

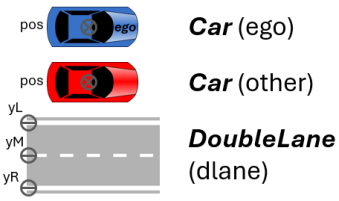
the symbols on the canvas, spatial relations of objects can be expressed via predicate logic formulae. The anchors form the basis for further elements such as distance lines (constraining the distance between two objects), nowhere-boxes (excluding object types in a specific area), or somewhere-boxes (defining a specific area in which an object can be). Predicate logic formulas for TSC specifications can be derived by the algorithms presented in [15]. In the following, we provide a detailed description of both the visual and formal specification of the functional described traffic situation, i.e., our exemplary context (see Ex. 1).

4.2 Specifying Traffic Situation $S_i \in \mathbf{S}$

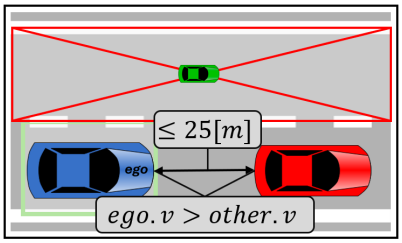
The specification of our exemplary traffic situation as an abstract traffic situation using TSC is shown in Fig. 2. It defines the described traffic situation with a *Basic Chart* (Fig. 2c) over an *OM* (Fig. 2a)

Class	Attribute	Basic Type	Domain
DoubleLane			
	yR	yposition	$\mathbb{R}$
	yM	yposition	$\mathbb{R}$
	yL	yposition	$\mathbb{R}$
Car			
	pos	position	$\mathbb{R}^2$
	v	velocity	$\mathbb{R}$
	inRange(*left*)	function	$\{i \mid i \in Car\}$

(a) Object model *OM* defining two classes, their attributes according to our example.



(b) Symbol Dictionary defining object symbols and anchors according to our example.



(c) Basic Chart describing the spatial relations and constraints on object attributes according to our example.

Figure 2: TSC Basic Chart of the example with an object model *OM* (a), a corresponding symbol dictionary (b), and an invariant node encapsulating one spatial view (c).

and a symbol dictionary (Fig. 2b). The *AV* (blue car symbol, named *ego*) is placed left of the slower driving (constraint on velocity between both symbols) car (red car symbol, named *other*) on the 2D canvas, expressing that *ego* is driving towards *other*. The described distance between *ego* and *other* is expressed by a *distance line* connecting symbol anchors and annotated with the distance constraint. Both car symbols have one anchor, while the two-lane road has three anchors for the three lane boundaries. This enables the specification of all described spatial relations. A so-called somewhere-box around *ego* and *other* defines that any lateral position of the cars in their lane (between *DoubleLane*'s anchors *yR* and *yM*) is valid. A nowhere-box with a car symbol on the left lane (between *DoubleLane*'s anchors *yM* and *yL*) defines the described situation, that there is no other car on the adjacent left lane.

In addition, the *OM* has a function called *inRange()*. The nowhere-box can be mapped to this function so that *ego* can determine how many other objects of the class *Car* exist in the range (e.g., adjacent lane sensor range of a real car's lane change assistant (LCA)) of *ego* in the left lane.

The derived predicate logical formula (supported by the algorithms presented in [15]) for the specification in Fig. 2 is shown in Eq. 1. In Eq. 1a, the spatial relation between *ego* and the *DoubleLane* is defined over *OM*. In Eq. 1b the spatial relation between the *other* car and the *DoubleLane* is defined. Eq. 1c expresses the distance constraint between *ego* and *other*, and Eq. 1d the

$$B1 = ego.pos.y > lane.yR \wedge ego.pos.y < lane.yM \quad (1a)$$
$$\wedge\, other.pos.y > lane.yR \wedge other.pos.y < lane.yM \quad (1b)$$
$$\wedge\, ego.pos.x + 25 \leq other.pos.x \quad (1c)$$
$$\wedge\, ego.v > other.v \quad (1d)$$
$$\wedge\, ego.inRange(left) = \varnothing \quad (1e)$$

difference in velocity. Eq. 1e expresses the constraint for the (car) empty adjacent left lane of *ego*.

By specifying the exemplary traffic situation as an abstract traffic situation S using TSC, a whole set of concrete situations is covered. Thus, in any concrete situation satisfying the abstract traffic situation, the same explanation will be applied. This allows for capturing infinitely many traffic situations with a finite number of explanations.

The formalisation of a traffic situation as TSC formalises the context C, which is important for the interpretation, traceability, and comprehensibility of an explanation E. Such traffic situation specification S also serves as the formal specification for the construction of a TSC runtime monitor. In Section 5 we describe our use of TSC runtime monitoring for recognising the entering or being of AV in a concrete traffic situation σ, which corresponds to traffic situation specification S, i.e., $\sigma \vDash S$. Hence, S is a formal foundation for context-aware and ante-hoc presentation of any explanations for context C.

5 Context-Recognition and Ante-hoc Explanation at Runtime

To enable context-aware, ante-hoc explanations of the behaviour of an AV, we need to recognise which concrete traffic situation σ the AV is currently in. The goal is to trigger the presentation of a context-aware explanation E, corresponding to the traffic situation σ. We use the term *ante-hoc explanation* (of (un)expetable driving manoeuvres) to denote explanations that are presented before an AV behaves as explained. The term ante-hoc explanations is often used in the context of white-box AI models. However, here we use the term for explanations stemming from explanation models, that are designed using system design artefacts and stakeholder needs. Providing such explanations requires rapid recognition of the AV entering or being in a context C, as well as timely presentation. With this work, we contribute the technical capability for context-aware, ante-hoc explanations. Aspects such as optimal presentation duration or alignment with passengers' cognitive load [7] are beyond this work's scope. In Section 7, we demonstrate the feasibility of providing context-aware, ante-hoc explanations of (un)expectable manoeuvres at runtime.

In Section 4, we showed how a described context C of an explanation E can be formalised as abstract traffic situations S using TSC. Such a specification defines spatial relations and constraints for a set of object types, attributes, and predicates over a shared OM.

To determine whether an AV is currently in a concrete traffic situation σ corresponding to a TSC context specification S, we use runtime monitoring. A dedicated TSC runtime monitoring for scenario-based testing has been developed previously [32, 20]. Here, we adapt it to enable context-recognition and trigger ante-hoc explanations. Concretely, we check at runtime whether $\sigma \vDash S$ and, if so, present the corresponding explanation E.

5.1 Matching Concrete Situations with Context Specifications

Let σ denote a concrete traffic situation, provided from sensor data at runtime. We assume the existence of a valuation function ρ that maps each traffic object identity $id \in ID$ to its observed attribute values $\rho(id)(a)$, where $a \in attr(C)$ and $C \in OM$. Given the specification of S as an abstract traffic situation using TSC, such a specification can be interpreted as a set of well-typed logical predicates $P(S)$ over OM (see Eq. 1). A concrete traffic situation σ corresponds to a specification S if $\sigma \vDash S \iff \rho \vDash P(S)$, i.e., the current traffic environment and the states of all participating agents satisfy the constraints defined by the predicates in S under the valuation ρ.

This evaluation relies on a semantic interface that maps the AV to a runtime monitor based on S.

Specifically, sensor data and other relevant information (e.g., from V2X communication) must be mapped in a type-consistent way to the attributes defined in the *OM* used by S. For each object attribute in Eq. 1 (e.g., *ego.velocity* in m/s), an appropriate input from the *AV*'s available data must be provided according to the corresponding *OM* (see Fig. 2a). Only then can the runtime monitor deliver a meaningful verdict on whether $\sigma \vDash S$.

5.2 Monitor Verdicts for Explanation Selection

We first consider a single traffic situation specification S. For this specification, we define a runtime monitor $\mathcal{M}_S$ that evaluates whether the concrete traffic situation σ where *AV* is in, corresponds to S:

$$\mathcal{M}_S(\sigma) := \begin{cases} \top & \text{if } \sigma \vDash S \\ \bot & \text{otherwise} \end{cases} \tag{2}$$

Since an explanation model *EM* describes an *AV*'s behaviour across many traffic situations, it requires multiple explanations E_i with their respective contexts C_i, formalised as TSC specifications S_i. We therefore construct a monitoring system comprising a runtime monitor $\mathcal{M}_{S_i}$ for each $S_i \in \mathbf{S}$, where $\mathbf{S}$ is the set of all traffic situation specifications defined for *EM*.

We define a function $\mathcal{V}$ that, given the current traffic situation σ and a set of explanations $\mathbf{E}$, returns all explanations $E_i \in \mathbf{E}$ whose monitors $\mathcal{M}_{S_i}$ evaluate to $\top$:

$$\mathcal{V}(\sigma, \mathbf{E}) := \begin{cases} \{E_i \in \mathbf{E} \mid \mathcal{M}_{S_i}(\sigma) = \top\} & \text{if } \exists E_i \in \mathbf{E} : \mathcal{M}_{S_i}(\sigma) = \top, \\ \varnothing & \text{otherwise.} \end{cases} \tag{3}$$

The function thus returns all explanations which correspond to σ, or $\varnothing$ if none correspond. Having multiple explanations apply to the same situation can be useful, e.g., during development. To avoid ambiguities for other stakeholders, such as passengers, $\mathbf{E}$ can be required to be *well-formed*, i.e., $\forall E_i, E_j \in \mathbf{E}, i \neq j : \neg\exists\sigma\,(\mathcal{M}_{S_i}(\sigma) = \mathcal{M}_{S_j}(\sigma) = \top)$. For TSC specifications, this can be verified through consistency analysis introduced in [10].

If all runtime monitors $\mathcal{M}_{S_i}(\sigma)$ return $\bot$, two cases may occur:

(i) *AV* is in a traffic situation σ for which no context specification S_i was defined at design time, e.g., an unconsidered or potentially unsafe scenario.

(ii) The traffic situation σ is covered by *AV*'s explanation model *EM*, but the set of explanations $\mathbf{E}$ is incomplete, missing an E_i for this context.

This distinction is important for validation and trust: case (i) can guide extensions of the explanation model, while case (ii) reveals gaps in the existing explanations. Additionally, a $\bot$ verdict can trigger runtime data logging, indicate sensor or perception issues, or activate a *safety fallback* in unsafe, uncovered situations.

In conclusion, the goal is that for every concrete traffic situation σ that *AV* might encounter and is described in an explanation model *EM*, there exists an $E_i \in \mathbf{E}$ for each $\sigma \vDash S_i$. If this is not the case, the $\bot$ verdict acts as a meaningful diagnostic property for incompleteness of explanations, system failures, or domain violations.

With the presented runtime monitoring based on traffic situation specifications S_i linked to explanations E_i, we can recognise at runtime whether an *AV* is in a situation requiring an explanation. This enables the context-aware presentation of (ante-hoc) explanations, in this work concretely about (un)expectable driving manoeuvres. Recall that combining TSC-based context C formalisation and a dedicated runtime monitoring can be used to trigger any explanations. Regarding the MAB-EX Framework [11], this combination instantiates the ***M**onitoring* and ***A**nalyse* phase for triggering context-aware explanations at system runtime.

6 Specification of (Un)expectable Driving Manoeuvres

In the following, we present the specification of driving manoeuvres with Traffic Sequence Charts (TSC), that can be used to specify ante-hoc behaviour explanation of *AVs*. We define a *driving manoeuvre* as a purposeful change in the state of attributes of *AV* over time, taking into account the environment and other traffic participants. We define the annotation *expectable* and *unexpectable* to a driving manoeuvre as follows.

Definition 1 ((Un)expectable Driving Manoeuvre)**.** *Given a concrete traffic situation σ, we call an evolution $\pi : \mathbb{T} \to ID \to attr(\mathbf{C}) \to \mathcal{D}$ starting in $\sigma = \pi(0)$ an expectable (resp. unexpectable) driving manoeuvre if and only if, AV may (resp. may not) continue from σ with π.*

As introduced in Section 1, the term *(un)expectable* is used to describe manoeuvres that are permitted or not according to an *AV* 's explanation model. Unlike strict requirement compliance, this notion captures model-based behaviours while considering domain and AI-based uncertainties.

Referring to our exemplary context *C* of an *AV* (see Ex. 1), a functional description of an expectable and an unexpectable driving manoeuvre can be as follows:

Example 2 ((Un)expectable Driving Manoeuvres)**.** *Given this close distance between AV and the other vehicle ($\leq$ 25 meters) and a free adjacent left lane, an expectable manoeuvre is to initiate an overtaking by activating the left indicators and changing lanes. At least in Germany, an unexpectable driving manoeuvre is to initiate an overtaking on the right side (as this violates the traffic rules).*

Hence, a traffic scenario specification is also a (valid) combination of consecutive traffic situation specifications S_i, where each S_i specifies, e.g., a phase of a driving manoeuvre.

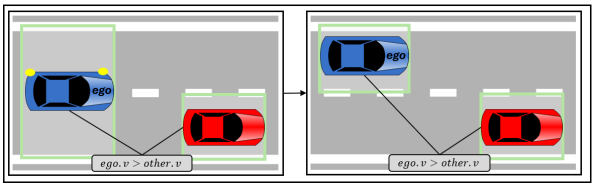
The specification in TSC provides a formalised and context-aware explanation $E := (S, \mathcal{F}_E, \mathcal{F}_V)$ of (un)expectable driving manoeuvrers, where S is the traffic situation specification, i.e. the context C of E, $\mathcal{F}_E$ is a non-empty set of specified *expectable* driving manoeuvrers, and $\mathcal{F}_V$ is a non-empty set of specified *unexpectable* driving manoeuvrers.

6.1 Visual and Formal Specification of Driving Manoeuvres using TSC

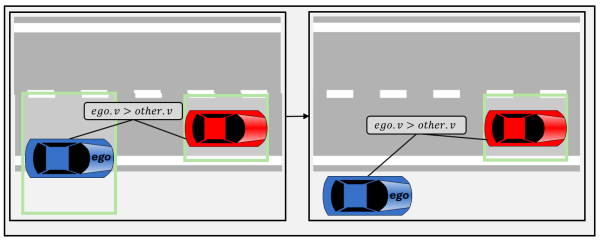
In TSC, *invariant nodes* can be connected consecutively over the same *OM* and same *symbol dictionary* (for a brief TSC introduction see Section 4.1). This enables the connection of traffic situation specifications resulting in abstract traffic scenario specifications. Such specifications can formally represent driving manoeuvres, satisfying our goal of specifying (un)expectable behaviour of *AVs*.

By applying available operators of TSC such as sequence, negation, choice or concurrency, invariant nodes can be composed into more complex chart structures (compared to a *Basic Chart*). When composed using the time model $\mathbb{T}$ of TSC, it allows the specification of abstract traffic scenarios, i.e. structures that semantically combine sets of well-typed predicate logic formulae over a given *OM* using the above operators. Formally, these scenarios can be regarded as expressions using operators over sets Φ_i of well-typed predicate logic formulae, i.e. expressions of the form $\mathcal{O}(\Phi_1, \Phi_2, \ldots, \Phi_n)$, where each Φ_i is a set of formulas over *OM* and $\mathcal{O}$ denotes an n-ary operator from the set of available operators in TSC. For this work, we only consider sequences of invariant nodes. The conditions of each invariant node N_i (such as Eq. 1) are conjugated. The semantics of a sequence chart *SC* containing two nodes N_1 and N_2 is defined as $[b,e] \vDash SC \iff \exists t \in [b,e] : [b,t] \vDash N_1 \wedge [t,e] \vDash N_2$, where $[b,e]$ is the time interval of evaluation. Further details on composed charts can be found in [15, 32]. Fig. 3a depicts a sequence chart with two invariant nodes.

In the first phase, a car changes to the left lane behind another car. The second phase completes the lane change. This sequence specifies our exemplary expectable manoeuvre. Similarly, Fig. 3b shows



(a) TSC Sequence Chart specifying an exemplary expectable manoeuvrer $F_i \in \mathcal{F}_E$ of *AV* initiating an overtaking of the other vehicle on the adjacent left lane.



(b) TSC Sequence Chart specifying an exemplary unexpectable manoeuvrer $F_i \in \mathcal{F}_V$ of *AV* initiating overtaking of the other vehicle on the right side.

Figure 3: Formalisation of exemplary (un)expectable manoeuvres for our context-aware explanation $E := (S, \mathcal{F}_E, \mathcal{F}_V)$, specified as abstract traffic scenarios using Traffic Sequence Charts [15].

a lane change to the right and its completion, representing our exemplary unexpectable manoeuvre. Both charts use the same *OM* and *symbol dictionary* as Fig. 2, with added boolean attributes for ego's indicators and corresponding visual features (yellow blobs).

Predicate logical formulas of the specifications in Fig. 3 can be derived with the support of algorithms in [15]. For brevity, and because we focus on the visual part of TSC-based manoeuvres as explanation means M at runtime, only the visual representations are shown. To illustrate the formal semantics of one invariant node N_i, we provide predicate logical formulas for two context specifications in detail (see Eq. 1 and Fig. 4), as needed for constructing runtime monitors (see Section 5). For completeness, the formal semantics for Fig. 3 can be found in the appendix (see Tab. 3).

According to our concept, an explanation E is a tuple $E := (S_i, \mathcal{F}_E, \mathcal{F}_V)$, where $S_i \in \mathbf{S}$ specifies the context C and explanandum X, $\mathcal{F}_E$ is the set of expectable manoeuvres, and $\mathcal{F}_V$ is the set of unexpectable manoeuvres, defined by the AV's explanation model EM. By specifying our exemplary context C (see Fig. 2) and the (un)expectable manoeuvres (see Fig. 3), we have all components for an explanation E using the visual yet formal TSC language. Given that $\mathcal{F}_E$ and $\mathcal{F}_V$ are correctly associated with $S_i \in \mathbf{S}$ in the explanation model EM of AV, the explanation $E = (S_i, \mathcal{F}_E, \mathcal{F}_V)$ becomes context-aware. Using the same *OM* and symbol dictionary, the abstract traffic situation specification S_i of the explanation context C determines *when* an explanation is required, while the associated manoeuvre sets $\mathcal{F}_E$ and $\mathcal{F}_V$ define *what* is to be communicated. In Section 7, we show a possible way of presenting such an explanation E.

In contrast to common XAI approaches, our method enables consistent alignment of context and explanation content already during design. This may benefit a variety of stakeholders: Testing engineers can validate that decision-making in specific traffic situations aligns with the formalised (un)expectable manoeuvres. Passengers or safety assessors may be informed why a specific manoeuvre is being (or not being) executed, increasing transparency and trust.

By combining explanation context and content specification using TSC with a dedicated runtime monitoring, we operationalise the loop between formal specification, runtime monitoring, and context-aware explanations, as conceptualised by the MAB-EX framework [11]. Hence, this approach supports the development of trustworthy autonomous driving systems.

7 Application of Context-aware, Ante-hoc TSC-Explanations

In this section, we demonstrate the applicability of our approach for presenting context-aware, ante-hoc explanations of (un)expectable driving manoeuvres and validate its runtime capability. Concretely, we specify multiple explanations E_i using TSC. Each explanation comprises an explanation context C_i,

formalised as an abstract traffic situation $S_i \in \mathbf{S}$, together with the corresponding (un)expectable driving manoeuvres ($\mathcal{F}_E$ and $\mathcal{F}_V$). We then construct runtime monitors $\mathcal{M}_{S_i}(\sigma)$ for each context C_i and embed them into a simulation. Finally, we provide a visual yet formal, ante-hoc presentation of the (un)expectable manoeuvres. Our approach is demonstrated in a simulated overtaking scenario of an *AV*.

To this end, Section 7.1 describes the experimental setup. In Section 7.2, we present the specified explanations E_i, each linking an abstract traffic situation S_i to individual phases of an overtaking manoeuvre: approaching, closing the gap, changing lanes, and overtaking. Section 7.3 then shows the runtime monitoring results and the ante-hoc explanations of (un)expectable manoeuvres. Finally, Section 7.4 discusses our method.

7.1 Experimental Setup

The hardware specification is reported to ensure reproducibility, comparability, and to provide a context for later runtime results. The hardware consists of 192 GB of RAM, an Intel® Xeon® Silver 4215R CPU (16 cores @ 3.20 GHz), and an NVIDIA® Quadro RTX 6000 GPU. CARLA [16] (version 0.9.15) was used as simulation environment for simulating the overtaking scenario. In the simulated traffic scenario, an *AV* (car, red) drives towards a slower-moving vehicle (van, grey) in the right-hand lane of a two-lane carriageway. At a certain distance, the *AV* initiates an overtaking manoeuvre, passes the vehicle to be overtaken and initiates the final lane change.

The runtime monitors were implemented in Python and connected to the CARLA simulation via the ROS2–CARLA Bridge [14]. Data were continuously transmitted at a frequency of 20 Hz during simulation runtime. We do not use CARLA's Python API directly, but rather ROS 2 as the communication protocol for *AV* data, as many real-world systems utilise ROS 2 for data transmission. This has enabled TSC runtime monitoring to be successfully deployed in a maritime context [6], directly from the simulation onto a real vessel.

Since our runtime monitors are constructed from abstract traffic situation specifications $S_i \in \mathbf{S}$ defined using TSCs, each monitor relies on the corresponding *OM* of the TSC specification. The object type instances and their respective attributes defined in the *OM* were mapped via an interface to suitable, available data provided by the CARLA simulation. Recall that only through appropriate mapping of objects and their attributes to suitable system data and signals can runtime monitoring deliver meaningful verdicts. Once a runtime monitor yields $\mathcal{M}_{S_i}(\sigma) = \top$ (see Eq. 2), indicating that the *AV* is in the corresponding traffic situation σ, the related explanation E_i for expectable ($\mathcal{F}_E$) and unexpectable ($\mathcal{F}_V$) manoeuvres is presented. The function $\mathcal{V}(\sigma, \mathbf{E})$ (see Eq. 3) that returns context-aware explanations for the concrete traffic situation σ of the *AV* was implemented in Python. It selects and triggers the presentation of corresponding explanations E_i. A video of the case study is provided in [21].

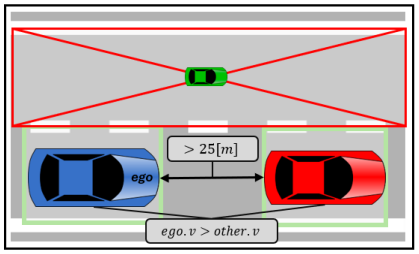
7.2 Defined Explanations

To demonstrate and validate our context-aware, ante-hoc explanation method, we decomposed the simulated overtaking into distinct phases. Each phase describes a context and its corresponding (un)expectable manoeuvres, which are visually and formally specified using TSC. Hence, the specification of each phase serves as our explanation in a tuple $E = (S_i, \mathcal{F}_E, \mathcal{F}_V)$. The following distinct phases of the considered overtaking are incorporating our example introduced earlier in Section 4 and Section 6 (here *Phase 2 - Closing Gap*):

Phase 1 - Approaching. *The AV is driving towards a slower driving vehicle in its lane. There is no other vehicle in the adjacent left lane of ego. The distance between ego and the other vehicle is greater than 25 metres. Given the moderate distance in this traffic situation, an expectable manoeuvres are (i) to initiate an overtaking by activating the left indicators and changing lanes, or (ii) to adapt to the speed of*

the leading vehicle and stay behind at a safe distance. An unexpectable driving manoeuvre is to initiate an overtaking on the right side (e.g., because of violating the traffic rules of German traffic law).

The specification of the abstract traffic situation S_i for *Phase 1* as TSC, as well as the derived predicate logical formula, is depicted in Fig. 4.



(a) Specified Basic Chart

$$
\begin{aligned}
S_1 = & ego.pos.y > lane.yR \wedge ego.pos.y < lane.yM \\
& \wedge other.pos.y > lane.yR \wedge other.pos.y < lane.yM \\
& \wedge ego.pos.x + 25 > other.pos.x \\
& \wedge ego.v > other.v \\
& \wedge ego.inRange(left) = \varnothing
\end{aligned}
$$

(b) Corresponding predicate logical formula

Figure 4: TSC specification of described traffic situation of *AV* in *Phase 1 - Approaching*, using the object model *OM* Fig. 2a and symbol dictionary Fig. 2b

The visual specifications of described (un)expectable driving manoeuvres for *Phase 1* are depicted in Fig. 5.Readers interested in the formal semantics of the visual specifications are referred to the appendix (seeTab. 2).

For presentation at runtime, we embedded the specifications of driving manoeuvres in a colored frame. This is just one possible way of presenting our explanations.

We chose a blue frame to communicate the case of more than one expectable driving manoeuvre. If there is only one expectable driving manoeuvre, we choose a green colored frame. We choose a red frame for every unexpectable driving manoeuvre.

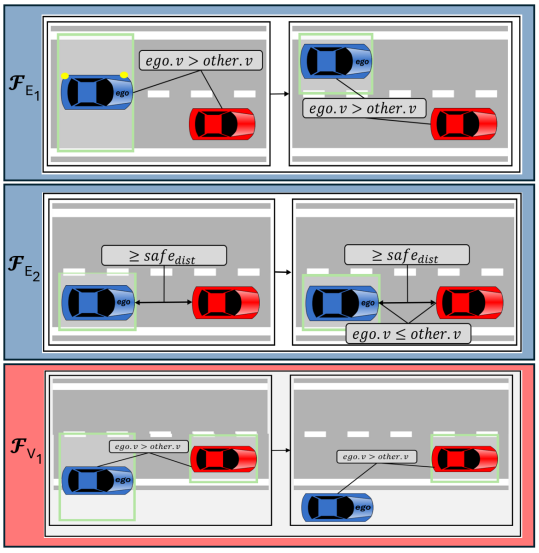


Figure 5: Representation of (un)expectable driving manoeuvres ($\mathcal{F}_{E_1}, \mathcal{F}_{E_2}, \mathcal{F}_{V_1}$) of an explanation $E_i \in \mathbf{E}$ for *Phase 1 - Approaching* (see Sec. 7.2), specified as abstract traffic scenarios using TSC [15].

Phase 2 - Closing Gap. *AV is following a slower driving vehicle (slower than speed limit) at a distance ≤ 25 meters. Given this close distance between AV and the other vehicle (≤ 25 meters) and a free adjacent left lane, an expectable manoeuvre is to initiate an overtaking by activating the left indicators and changing lanes. An unexpectable driving manoeuvre is to initiate an overtaking on the right side (e.g., because of violating the traffic rules in the German StVO).*

The specified explanation E_2 of *Phase 2* using TSC has already been conducted in Section 4. The colour-coded version for the presentation in simulation of the (un)expectable driving manoeuvres is depicted in Tab. 1 Readers interested in the formal semantics of the visual specifications are referred to the appendix (see Tab. 3).

Phase 3 - Lane Change. *AV is initiating an overtaking of a slower driving vehicle (slower than speed limit) by changing lanes with active left indicators. An expectable driving manoeuvre is to reach the*

adjacent left lane and begin passing the slower-moving vehicle. An unexpectable driving manoeuvre, given to the free target lane, is to abort the lane change, driving slower, and then driving behind the other vehicle at a safe distance.

The explanation context of *Phase 3*, specified as abstract traffic situation S_3, along with the colour-coded explanation E_3 showing the (un)expectable manoeuvres, is depicted in Tab. 1. Readers interested in the formal semantics of the visual specifications are referred to the appendix (see Tab. 4).

Phase 4 - Overtaking. *AV is driving on the left lane of a two-lane carriageway and is about to overtake a slower driving vehicle (slower than the speed limit) on the right lane. An expectable driving manoeuvre is to initiate a lane change to the right lane and in front of the vehicle to be overtaken. An unexpectable driving manoeuvre is to drive slower than the other vehicle and to abort the overtaking by a lane change to the right behind the other vehicle.*

The explanation context of *Phase 4*, specified as abstract traffic situation S_4, along with the colour-coded explanation E_4 showing the (un)expectable manoeuvres, is depicted in Tab. 1. Readers interested in the formal semantics of the visual specifications are referred to the appendix (see Tab. 5).

7.3 Runtime Detection and Explanation Timing

For runtime monitoring in the simulation, the class instances and attributes from the defined *OM* (Fig. 2a) were directly mapped to available simulation signals. A reliable interface between vehicle sensor data and runtime monitors is essential: We converted measurements (e.g., bounding boxes, speeds) into typed *OM* attributes (position $\in \mathbb{R}^2$, velocity $\in \mathbb{R}$, lane ID, etc.). Corresponding to the four overtaking phases, we constructed four runtime monitors, each evaluating a predicate formula $P(S_i)$ derived from TSC specifications of contexts C_i. When $\rho \vDash P(S_i)$, the associated explanation E_i is selected and presented ante-hoc, i.e., before execution. For each $E_i \in \mathbf{E}$, the corresponding runtime monitor $\mathcal{M}_{S_i}(\sigma)$ is embedded in the simulation, as detailed in Section 7.1.

Fig. 6 visualises the evolution of relevant attributes in the simulated overtaking over time. It also outlines at what point in time which monitor $\mathcal{M}_{S_i}(\sigma)$ recognises a corresponding concrete traffic situation σ, i.e. $\mathcal{M}_{S_i}(\sigma) = \top$ (blue-ish/grey-ish frames). The indexing of the monitors corresponds to the phases described, i.e. $\mathcal{M}_{S_1}(\sigma)$ is the runtime monitor constructed from the abstract traffic situation specification S_1 and *Phase 1 - Approaching* (see Fig. 4).

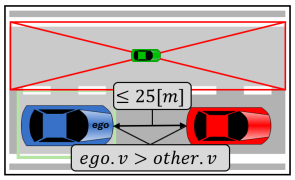
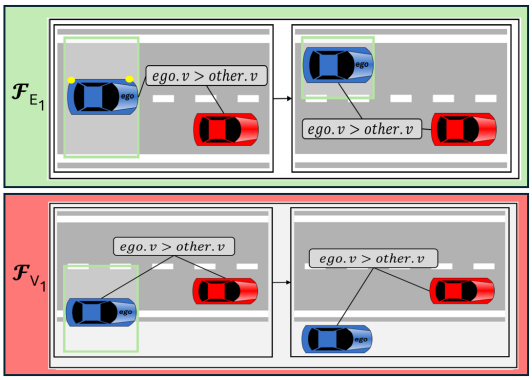
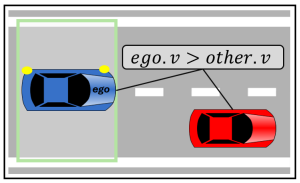
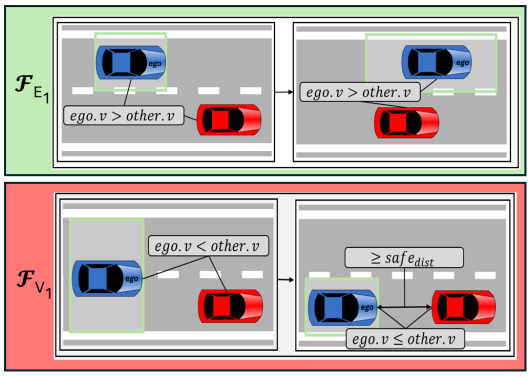
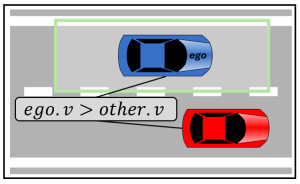
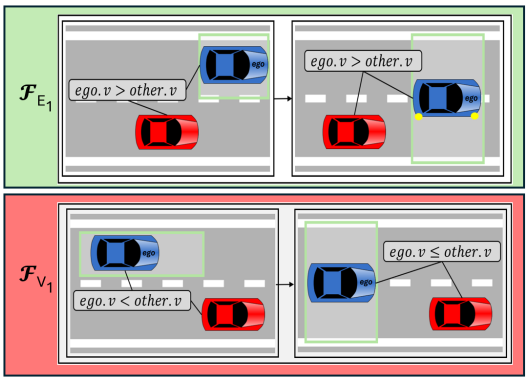
The runtime monitors $\mathcal{M}_{S_i}(\sigma)$ were able to recognise the corresponding explanation contexts, i.e., evaluate $\sigma \vDash S_i$, in under 1 ms after the *AV* entered the respective traffic situation σ in the simulation. This shows that TSC-based runtime monitoring can reliably detect $\sigma \vDash S_i$. It also enables the context-aware presentation of explanations, effectively operationalising the *Monitoring* and *Analysis* phases of the MAB-EX framework [11].

Fig. 6 also shows which explanation E_i was presented and for how long. Since each explanation E_i for an abstract traffic situation S_i is correctly associated with its (un)expectable driving manoeuvres, it demonstrates that the context-aware explanation E_i was always presented when the *AV* was in a corresponding concrete traffic situation $\sigma \vDash S_i$. Hence, this demonstrates the operationalisation of phases *Build* and *EXplain* of the MAB-EX framework [11]. Since the respective explanation E_i was presented immediately after $\mathcal{M}_{S_i}(\sigma) = \top$, and every respective traffic situation is present for more than 2 seconds in the simulation, our explanation of (un)expectable driving manoeuvres is ante-hoc.

7.4 Discussion

Our demonstration validates in a simulated overtaking that our approach is able to provide (i) context-aware, ante-hoc triggering of explanations, and (ii) visually yet formally specified (un)expectable driving

Table 1: Overview of context specifications C_i and corresponding (un)expectable driving manoeuvres ($E_i \in \mathbf{E}$) for the consecutive phases 2-4 of the overtaking scenario. Each context is specified as a TSC-based traffic situation specification. The corresponding explanation is specified as an abstract traffic scenario according to the object model *OM* (Fig. 2a) and symbol dictionary (Fig. 2b).

Phase	**Context**	**(Un)expectable Manoeuvrers**
Phase 2 – Closing Gap		
Phase 3 – Lane Change		
Phase 4 – Overtaking		

manoeuvres before an *AV* is manoeuvring. The explanations were specified in a visual yet formal manner using TSC, designed to be context-aware, and combined with a dedicated runtime monitoring.

By implementing our approach in the simulation environment CARLA, we have demonstrated its capabilities. Explanations can be triggered ante-hoc (before a manoeuvre is executed), based on the concrete traffic situation σ in which the *AV* is currently in and which corresponds to a formal explanation

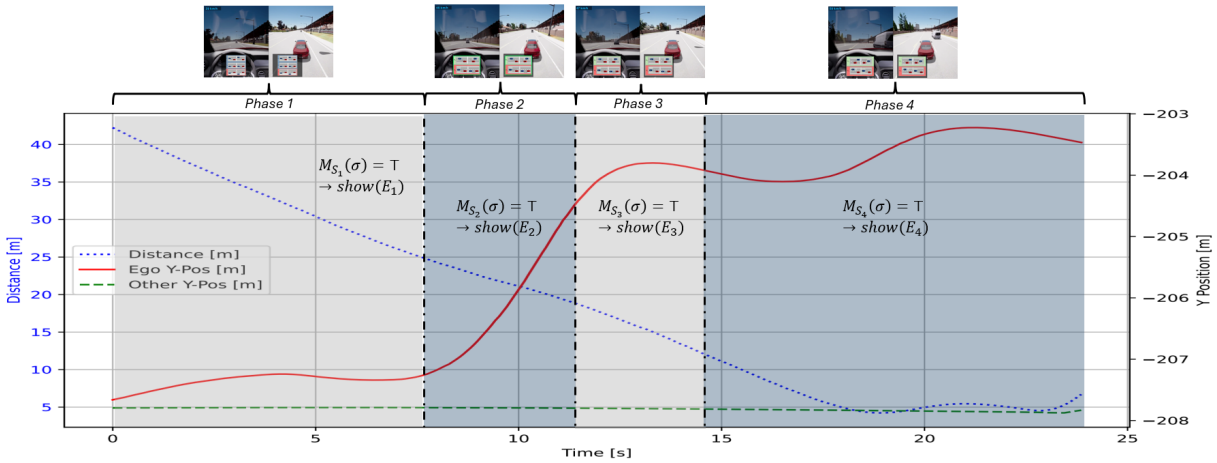


Figure 6: Important attributes (distance between *AV* (ego) and the van (other), blue graph; y-position of *AV* (ego), red graph; and y-position of the van (other), green graph) of the simulated overtaking scenario presenting the decomposed phases with images of the simulation above and length, marked with brackets over the time dimension. The blue-ish and grey-ish frames describe the time at which the respective monitors $\mathcal{M}_{S_i}(\sigma)$ recognised a corresponding concrete traffic situation σ, as well as which explanation E_i was subsequently presented. The dashed vertical lines describe the time at which another runtime monitor recognises the corresponding traffic situation.

context specified as an abstract traffic situation S_i. Regarding the scalability of our approach, there is tool support for the creation of TSC specifications and the generation of predicate logic formulas [13]. Executing several runtime monitors in simulation is not a limiting factor. The demonstration was restricted to a controlled and simulated overtaking and did not include testing in more complex environments. The simulation setup also relied on idealised sensor data and environmental models, without accounting for factors such as sensor noise. Further, to investigate whether a TSC-based explanation is indeed able to fulfil the goal of increasing trust, empirical studies need to be conducted with stakeholder groups. Hence, for now, we can not provide any empirical statement if an explanation E specified by TSC is "*... a piece of information (or evidence) that makes the explanandum X understandable by G with respect to the goal* θ."[31]. However, we believe that the combination of visual syntax and rigorous formal semantics based on predicate logic allows the same TSC explanation to address different stakeholders. For example, the visual representation of E could address passengers of *AV*, while the formal semantics could address developers or test engineers.

8 Conclusion and Future Work

This work aims to increase trust in autonomous vehicles (*AVs*) by enabling context-aware, ante-hoc explanations at system runtime to support the bridging of correct (by rigorous formal semantics) and good (by both visual and formal means) explanations. Within the field of Explainability Engineering, we present a formalisation of explanation contexts, i.e., a traffic situation of an *AV* where an explanation is meant to be presented, using Traffic Sequence Charts (TSC). In combination with a dedicated TSC runtime monitoring, we can observe the current traffic situation of *AVs*, evaluate if it corresponds to the formalised explanation context, and trigger the presentation of explanations. Additionally, we show how TSC can formalise (un)expectable driving manoeuvres in a context-aware, visual yet formal way,

enabling the same explanation to be used for different stakeholders. In a simulated overtaking scenario, we demonstrate and validate our contribution of context-aware explanations of (un)expectable driving manoeuvres, and show that we were able to present ante-hoc explanations, i.e., before an *AV* is manoeuvring, at system runtime. This work aligns with the MAB-EX framework [11] and is the first concrete operationalisation of key phases that demonstrates the runtime applicability for context-aware, ante-hoc explanations.

To enhance the applicability and impact of our approach, several extensions are envisaged. Future work may expand the approach to various and more complex driving scenarios, incorporating uncertainty-aware runtime monitoring techniques (cf. [19]). This may include empirical studies with diverse stakeholders, designing multimodal explanations (e.g., textual annotations or audio cues), and evaluating feasibility on automotive-grade hardware. Furthermore, we see a potential in combining TSC specifications of explanation contexts with the explanation history extension of the MAB-EX framework [31], as such abstract specifications can generalise over several explanation contexts.

References

[1] (2024): *Regulation (EU) 2024/1689 of the European Parliament and of the Council of 13 June 2024 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act)*. Available at `https://eur-lex.europa.eu/eli/reg/2024/1689/oj/eng`. Official Journal of the European Union, L 1689.

[2] Gulsum Alicioglu & Bo Sun (2022): *A survey of visual analytics for Explainable Artificial Intelligence methods*. *Computers and Graphics* 102, pp. 502–520, DOI: `10.1016/j.cag.2021.09.002`.

[3] Thaar Alqahtani (2025): *Recent Trends in the Public Acceptance of Autonomous Vehicles: A Review*. *Vehicles* 7(2), DOI: `10.3390/vehicles7020045`.

[4] Nikos Arechiga (2019): *Specifying Safety of Autonomous Vehicles in Signal Temporal Logic*. In: *2019 IEEE Intelligent Vehicles Symposium (IV)*, pp. 58–63, DOI: `10.1109/IVS.2019.8813875`.

[5] Zahra Atf & Peter R. Lewis (2025): *Is Trust Correlated With Explainability in AI? A Meta-Analysis*. *IEEE Transactions on Technology and Society*, pp. 1–8, DOI: `10.1109/TTS.2025.3558448`.

[6] Anna Austel, Lukas Panneke, Janusz Andrzej Piotrowski, Nina Wetzig, Matthias Steidel & Bernd Westphal (2025): *Using Monitoring of Maritime Traffic Scenarios in the Validation of Maritime Systems*. In: *2025 Symposium on Maritime Informatics and Robotics (MARIS)*, DOI: `10.1109/MARIS64137.2025.11139541`. Available at `https://elib.dlr.de/215965/`.

[7] A. Bairy & M. Fränzle (2023): *Optimal Explanation Generation using Attention Distribution Model*. In Tareq Ahram & Redha Taiar, editors: *Human Interaction and Emerging Technologies (IHIET-AI 2023): Artificial Intelligence and Future Applications*, *AHFE Open Access* 70, AHFE International, USA, DOI: `10.54941/ahfe1002928`.

[8] Ezio Bartocci, Yliès Falcone, Adrian Francalanza & Giles Reger (2018): *Introduction to Runtime Verification*. In: *Lectures on Runtime Verification. Introductory and Advanced Topics*, *Lecture Notes in Computer Science* 10457, Springer, pp. 1–33, DOI: `10.1007/978-3-319-75632-5_1`.

[9] Andreas Bauer, Martin Leucker & Christian Schallhart (2011): *Runtime Verification for LTL and TLTL*. *ACM Trans. Softw. Eng. Methodol.* 20(4), DOI: `10.1145/2000799.2000800`.

[10] Jan Steffen Becker (2024): *A Consistency Analysis Method for Traffic Sequence Charts*. In: *VEHITS24 Doctoral Consortium*, DOI: `10.48550/arXiv.2409.03774`. Available at `https://elib.dlr.de/204347/`.

[11] Mathias Blumreiter, Joel Greenyer, Francisco Javier Chiyah Garcia, Verena Klös, Maike Schwammberger, Christoph Sommer, Andreas Vogelsang & Andreas Wortmann (2019): *Towards Self-Explainable Cyber-Physical Systems*. In: *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*, pp. 543–548, DOI: `10.1109/MODELS-C.2019.00084`.

[12] Dimitri Bohlender & Maximilian A. Köhl (2019): *Towards a Characterization of Explainable Systems*. arXiv:1902.03096.

[13] Philipp Borchers, Tjark Koopmann, Lukas Westhofen, Jan Steffen Becker, Lina Putze, Dominik Grundt, Thies de Graaff, Vincent Kalwa & Christian Neurohr (2025): *TSC2CARLA: An abstract scenario-based verification toolchain for automated driving systems*. *Science of Computer Programming* 242, p. 103256, DOI: 10.1016/j.scico.2024.103256. Available at https://www.sciencedirect.com/science/article/pii/S0167642324001795.

[14] carla-simulator (2025): *carla_ros_bridge: ROS/ROS2 Bridge for CARLA Simulator*. https://github.com/carla-simulator/ros-bridge. Accessed: 2025-08-12.

[15] Werner Damm, Stephanie Kemper, Eike Möhlmann, Thomas Peikenkamp & Astrid Rakow (2017): *Traffic Sequence Charts - From Visualization to Semantics*. AVACS Technical Report (117), DOI: 10.13140/RG.2.2.15190.42563.

[16] Alexey Dosovitskiy, Germán Ros, Felipe Codevilla, Antonio M. López & Vladlen Koltun (2017): *CARLA: An Open Urban Driving Simulator*. In: *Conference on Robot Learning*. Available at https://api.semanticscholar.org/CorpusID:5550767.

[17] Na Du, Jacob Haspiel, Qiaoning Zhang, Dawn Tilbury, Anuj K. Pradhan, X. Jessie Yang & Lionel P. Robert (2019): *Look who's talking now: Implications of AV's explanations on driver's trust, AV preference, anxiety and mental workload*. *Transportation Research Part C: Emerging Technologies* 104, pp. 428–442, DOI: 10.1016/j.trc.2019.05.025.

[18] Klemens Esterle, Luis Gressenbuch & Alois Knoll (2020): *Formalizing Traffic Rules for Machine Interpretability*. In: *2020 IEEE 3rd Connected and Automated Vehicles Symposium (CAVS)*, pp. 1–7, DOI: 10.1109/CAVS51000.2020.9334599.

[19] Bernd Finkbeiner, Martin Fränzle, Florian Kohn & Paul Kröger (2022): *A Truly Robust Signal Temporal Logic: Monitoring Safety Properties of Interacting Cyber-Physical Systems under Uncertain Observation*. *Algorithms* 15(4), DOI: 10.3390/a15040126.

[20] Dominik Grundt, Anna Köhne, Ishan Saxena, Ralf Stemmer, Bernd Westphal & Eike Möhlmann (2022): *Towards Runtime Monitoring of Complex System Requirements for Autonomous Driving Functions*. In Matt Luckcuck & Marie Farrell, editors: 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, *Electronic Proceedings in Theoretical Computer Science* 371, Open Publishing Association, pp. 53–61, DOI: 10.4204/EPTCS.371.4.

[21] Dominik Grundt, Ishan Saxena, Malte Petersen, Bernd Westphal & Eike Möhlmann (2025): *Application Video: Simulated Overtaking*. https://www.youtube.com/watch?v=5rFGkbvKOjs. Supplementary video material illustrating the demonstration of our approach in a simulated overtaking.

[22] Klaus Havelund & Grigore Roşu (2002): *Synthesizing Monitors for Safety Properties*. In Joost-Pieter Katoen & Perdita Stevens, editors: *Tools and Algorithms for the Construction and Analysis of Systems*, Springer Berlin Heidelberg, Berlin, Heidelberg, pp. 342–356, DOI: 10.1007/3-540-46002-0_24.

[23] Robert A Kaufman, Aaron Broukhim, David Kirsh & Nadir Weibel (2025): *What Did My Car Say? Impact of Autonomous Vehicle Explanation Errors and Driving Context On Comfort, Reliance, Satisfaction, and Driving Confidence*. In: *Proceedings of the 2025 CHI Conference on Human Factors in Computing Systems*, CHI '25, Association for Computing Machinery, New York, NY, USA, DOI: 10.1145/3706598.3713088.

[24] Maximilian A. Köhl, Kevin Baum, Markus Langer, Daniel Oster, Timo Speith & Dimitri Bohlender (2019): *Explainability as a Non-Functional Requirement*. In: *2019 IEEE 27th International Requirements Engineering Conference (RE)*, pp. 363–368, DOI: 10.1109/RE.2019.00046.

[25] Shuai Ma (2024): *Towards Human-centered Design of Explainable Artificial Intelligence (XAI): A Survey of Empirical Studies*. arXiv:2410.21183.

[26] Kareem Othman (2021): *Public acceptance and perception of autonomous vehicles: a comprehensive review*. *AI Ethics* 1(3), pp. 355–387, DOI: 10.1007/s43681-021-00041-8.

[27] Guglielmo Papagni, Jesse de Pagter, Setareh Zafari, Michael Filzmoser & Sabine T. Koeszegi (2022): *Artificial agents' explainability to support trust: considerations on timing and context*. *AI Soc.* 38(2), p. 947–960, DOI: 10.1007/s00146-022-01462-7.

[28] Albert Rizaldi, Jonas Keinholz, Monika Huber, Jochen Feldle, Fabian Immler, Matthias Althoff, Eric Hilgendorf & Tobias Nipkow (2017): *Formalising and Monitoring Traffic Rules for Autonomous Vehicles in Isabelle/HOL*. In Nadia Polikarpova & Steve Schneider, editors: *Integrated Formal Methods*, Springer International Publishing, Cham, pp. 50–66, DOI: 10.1007/978-3-319-66845-1_4.

[29] Maike Schwammberger (2025): *From Explanation Correctness to Explanation Goodness: Only Provably Correct Explanations Can Save the World*. In Bernhard Steffen, editor: *Bridging the Gap Between AI and Reality*, Springer Nature Switzerland, Cham, pp. 307–317, DOI: 10.1007/978-3-031-73741-1_19.

[30] Maike Schwammberger & Verena Klös (2022): *From Specification Models to Explanation Models: An Extraction and Refinement Process for Timed Automata*. In Matt Luckcuck & Marie Farrell, editors: 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, *Electronic Proceedings in Theoretical Computer Science* 371, Open Publishing Association, pp. 20–37, DOI: 10.4204/EPTCS.371.2.

[31] Maike Schwammberger, Raffaela Mirandola & Nils Wenninghoff (2024): *Explainability Engineering Challenges: Connecting Explainability Levels to Run-Time Explainability*. In Luca Longo, Sebastian Lapuschkin & Christin Seifert, editors: *Explainable Artificial Intelligence*, Springer Nature Switzerland, Cham, pp. 205–218, DOI: 10.1007/978-3-031-63803-9_11.

[32] Ralf Stemmer, Ishan Saxena, Lukas Panneke, Dominik Grundt, Anna Austel, Eike Möhlmann & Bernd Westphal (2025): *Runtime monitoring of complex scenario-based requirements for autonomous driving functions*. *Science of Computer Programming* 244, p. 103301, DOI: 10.1016/j.scico.2025.103301.

[33] Cumhur Erkan Tuncali, Georgios Fainekos, Hisahiro Ito & James Kapinski (2018): *Simulation-based Adversarial Test Generation for Autonomous Vehicles with Machine Learning Components*. In: *2018 IEEE Intelligent Vehicles Symposium (IV)*, pp. 1555–1562, DOI: 10.1109/IVS.2018.8500421.

Annex

Table 2: Specified (un)expectable driving manoeuvres ($\mathcal{F}_{E_1}, \mathcal{F}_{E_2}, \mathcal{F}_{V_1}$) and their corresponding predicate logical formulas for *Phase 1 - Approaching* of the overtaking scenario. The manoeuvres are represented as abstract traffic scenarios using TSC [15], according to the object model *OM* Fig. 2a and symbol dictionary Fig. 2b (except that we added $indicators_{left}$ attribute for class *Car*, and a visual feature (yellow blobs) in the symbol dictionary for the indicators). The corresponding context C_1 and formula are introduced in Fig. 4. The semicolon separates the invariant nodes, indicating the temporal partitioning of the time interval $[b, e]$ over which the sequence chart is evaluated.

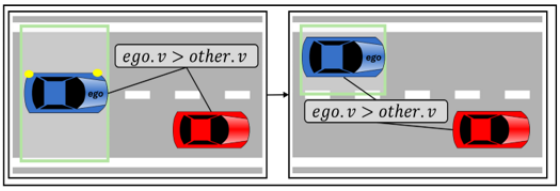
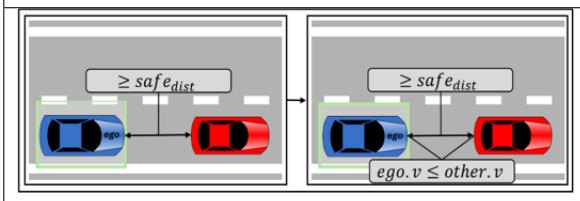
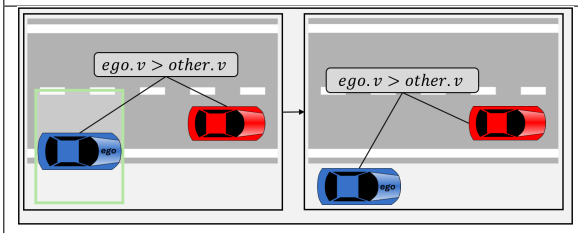
Specified Manoeuvrers	**Predicate Logical Formula**
	$F_{E_1} = ego.pos.y > lane.yR \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v$ $\wedge\ ego.indicators_{left} = 1;$ $ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v$
	$F_{E_2} = ego.pos.y > lane.yR \wedge ego.pos.y < lane.yM$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x + safe_{dist} < other.pos.x;$ $ego.pos.y > lane.yR \wedge ego.pos.y < lane.yM$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x + safe_{dist} < other.pos.x$ $\wedge\ ego.v \leq other.v$
	$F_{V_1} = ego.pos.y > lane.yM$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v;$ $ego.pos.y < lane.yR$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v$

Table 3: Specified (un)expectable driving manoeuvres ($\mathcal{F}_{E_1}, \mathcal{F}_{V_1}$) and their corresponding predicate logical formulas for *Phase 2 - Closing Gap* of the overtaking scenario. The manoeuvres are represented as abstract traffic scenarios using TSC [15], according to the object model *OM* Fig. 2a and symbol dictionary Fig. 2b (except that we added $indicators_{left}$ attribute for class *Car*, and a visual feature (yellow blobs) in the symbol dictionary for the indicators). The corresponding context C_2 and formula are introduced in Eq. 1. The semicolon separates the invariant nodes, indicating the temporal partitioning of the time interval $[b, e]$ over which the sequence chart is evaluated.

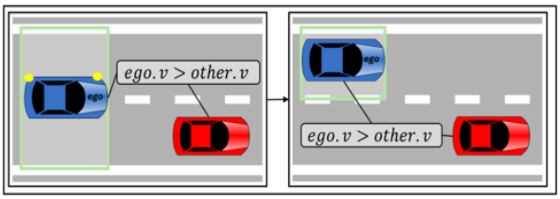
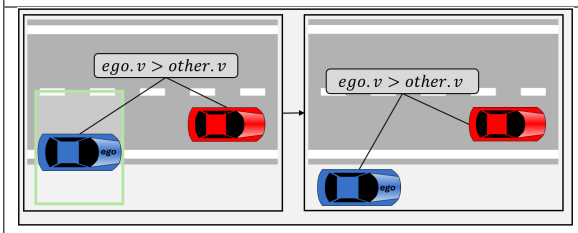
Specified Manoeuvrers	**Predicate Logical Formula**
	$F_{E_1} = ego.pos.y > lane.yR \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v$ $\wedge\ ego.indicators_{left} = 1;$ $ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v$
	$F_{V_1} = ego.pos.y < lane.yM$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v;$ $ego.pos.y < lane.yR$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v$

Table 4: Specified context C_3 and (un)expectable driving manoeuvres ($\mathcal{F}_{E_1}, \mathcal{F}_{V_1}$), and corresponding predicate logical formulas for *Phase 3 - Lane Change* of the overtaking scenario. The specifications are based on the object model *OM* Fig. 2a and symbol dictionary Fig. 2b (except that we added $indicators_{left}$ attribute for class *Car*, and a visual feature (yellow blobs) in the symbol dictionary for the indicators). The semicolon separates the invariant nodes, indicating the temporal partitioning of the time interval $[b, e]$ over which the sequence chart is evaluated.

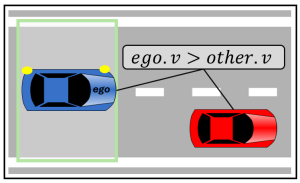
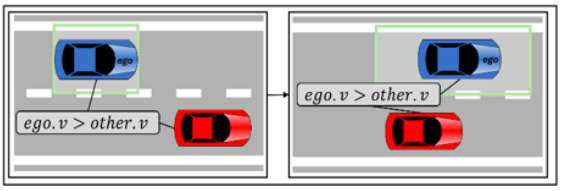
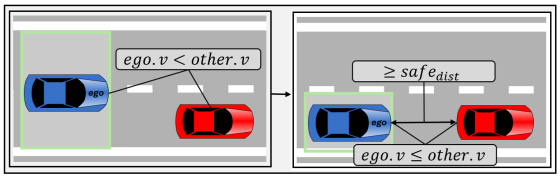
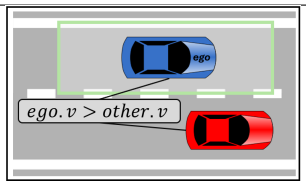
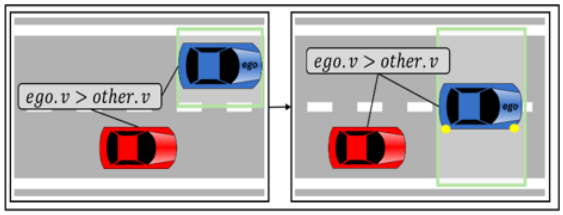
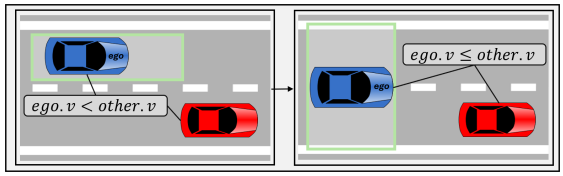
Specified Context & Manoeuvrers	**Predicate Logical Formula**
	$C_3 = ego.pos.y > lane.yR \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.indicators_{left} = 1$
	$F_{E_1} = ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v > other.v;$ $ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x \geq other.pos.x$ $\wedge\ ego.v > other.v$
	$F_{V_1} = ego.pos.y > lane.yR \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v < other.v;$ $ego.pos.y > lane.yR \wedge ego.pos.y < lane.yM$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x + safe_{dist} \leq other.pos.x$ $\wedge\ ego.v \leq other.v$

Table 5: Specified context C_4 and (un)expectable driving manoeuvres ($\mathcal{F}_{E_1}, \mathcal{F}_{V_1}$), and corresponding predicate logical formulas for *Phase 4 - Overtaking* of the overtaking scenario. The specifications are based on the object model *OM* Fig. 2a and symbol dictionary Fig. 2b (except that we added $indicators_{left}$ attribute for class *Car*, and a visual feature (yellow blobs) in the symbol dictionary for the indicators). The semicolon separates the invariant nodes, indicating the temporal partitioning of the time interval $[b, e]$ over which the sequence chart is evaluated.

Specified Context & Manoeuvrers	**Predicate Logical Formula**
	$C_4 = ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x \leq other.pos.x$ $\wedge\ ego.v > other.v$
	$F_{E_1} = ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x > other.pos.x$ $\wedge\ ego.v > other.v;$ $ego.pos.y > lane.yR \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x > other.pos.x$ $\wedge\ ego.v > other.v$ $\wedge\ ego.indicators_{right} = 1$
	$F_{V_1} = ego.pos.y > lane.yM \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v < other.v;$ $ego.pos.y > lane.yR \wedge ego.pos.y < lane.yL$ $\wedge\ other.pos.y > lane.yR \wedge other.pos.y < lane.yM$ $\wedge\ ego.pos.x < other.pos.x$ $\wedge\ ego.v \leq other.v$