Modeling and Analysing Cyber-Physical Systems in HOL-CSP

Paolo Crisafulli[1], Safouan Taha[2], and Burkhart Wolff[3]

[1] IRT SystemX, Palaiseau
paolo.crisafulli@irt-systemx.fr
[2] LMF, CentraleSupelec
safouan.taha@lri.fr
[3] LMF, Université Paris-Saclay
wolff@lri.fr

Abstract. Modelling and Analysing Cyber-Physical Systems (CPS) is a challenge for Formal Methods and therefore a field of active research. It is characteristic of CPSs that models comprise aspects of Newtonian Physics appearing in system environments, the difficulties of their discretisation, the problems of communication and interaction between actors in this environment as well as calculations respecting time-bounds. We present a novel framework to address these problems developed with industrial partners involved in the Autonomous Car domain. Based on HOL-CSP, we model time, physical evolution, “scenes” (global states) and “scenarios” (traces) as well as the interaction of “actors” (vehicles, pedestrians, traffic lights) inside this framework. In particular, discrete samplings are modelled by infinite internal choices.

For several instances of the modelling framework, we give formal proofs of a particular safety property for Autonomous Cars: if each car follows the same driving strategy defined by the so-called Responsibility-Sensitive Safety (RSS), no collision will occur. The proofs give rise to a number of variants of RSS and optimisations as well as a test-case partitioning of abstract test cases and a test-strategy for integration tests.

Keywords: Cyber-Physical Systems, Autonomous Cars, Safety-Critical Systems, Process-Algebra, Concurrency, Proof-based Verification

1 Introduction

As Cyber-Physical Systems (CPS) such as robots or Autonomous Vehicles (AV) are a developing industrial field, the need for safety certification is widening. Therefore, there is an important opportunity for formal methods to address this type of systems and a need to meet the respective scientific challenges. A widely acceptable definition of CPSs characterises them by:

- concurrent interaction of digitally controlled actors and an environment ...
- ... with continuous observables obeying the laws of Newtonian physics, and
- based on dense time $\mathbb{R}^+$ (or an even richer dense and compact structure).

Since we are interested in a refinement-based approach leading to concrete implementations of the actor-controllers, we would like to add the following two items to this list:

- actors possess only a discretised view of continuous observables (the world is sampled into "snapshots"), and
- computations in sensors and control systems typically have to meet hard time bounds.

Our understanding of "analysis" of CPS implies the establishment of their system properties by means of systematic formal test and mechanized proof. These techniques allow for stronger guarantees than *simulations* of scenarios which are widely used in the industrial practice albeit known limitations discussed in Sect. 6.1 in more detail.

In this paper, we present a *framework* developed in [11] for modeling and formally analysing CPS in general and concrete *instances* in the form of case studies from the AV domain. Roughly speaking, our framework is based on a conservative embedding of the theory of Communicating Sequential Processes (CSP) [25,7,38,42] into Isabelle/HOL [33] which serves both as modeling and analysis environment.

It turns out that CSP standard refinement notions can be adapted to meaningful notions in the CPS domain, and that in particular the discretized/sampled view of continuous variables lends itself to classical refinement reasoning. Finally, our approach paves the way to low level, machine oriented bitvector calculations in C programs verified by the Isabelle/AutoCorres environment [47,22].

This paper proceeds as follows: after an introduction to HOL-CSP, we present the general framework of interacting autonomous cars, which may be modeled in an object-oriented manner as in the domain-specific ontology languages such as MOSAR [4] or ASAM's OpenSCENARIO 2.0 [5]. The framework assumes AVs to be HOL-CSP processes called *actors* that agree on specific time points on a global physical state. (see Fig. 3) Based on this global physical state, a particular function inside an actor, the *driving strategy*, computes a set of possible accelerations. One of them is non-deterministically chosen and constantly executed in the time interval. The actor state will change in this time interval according to its kinematics.

Subsequently, inside this framework, we present a concrete topology and scenario with two or more cars on a lane, that follow the Responsibility-Sensitive Safety (RSS) principles defined in [40], providing a concrete formal definition of a *driving strategy* compatible with the law ("Duty of Care") and a paper-and-pencil proof trying to establish a safety property for it. RSS has been subject to numerous studies (see Sect. 6.1).

[4] `https://www.mosar.io`

[5] `https://asam-ev.github.io/public_release_candidate/asam-openscenario/2.0.0/welcome.html`

In the last part of this paper, we formalize, correct, and generalize this analysis by machine checked proofs — several of them are, to our knowledge, formally stated and analysed for the first time. Our formal analysis also allows for substantial improvements of the strategy (smaller safety distances by same safety goals) and work out ways to extract test data for system integration tests from the formal proofs.

2 Background

2.1 Classic CSP

The theory of CSP was first described in 1978 in a book by Tony Hoare [25], but has since evolved substantially [6,7,38]. The basic blocks of this *compositional* process language are *atomic events* e_1, e_2, ... from some set Σ, which were used to form the most basic processes via the prefix operator $e \rightarrow P$ (e happens, then the process P continues. Recursion $\mu X.\ P(X)$ allows for the description of infinite processes: $\mu X.\ e \rightarrow X$ describes the process that engages infinitely many times in the e event. CSP comes with a denotational semantics that can be described by projections: traces $\mathcal{T}$ and failures $\mathcal{F}$ [6]. The simplest one, the traces of $\mathcal{T}(\mu X.\ e \rightarrow X)$ map the process to the prefix-closed set of strings $\{[]$, $[e]$, $[e,e]$, ...$\}$ similarly to the language of regular expressions. There are two types of alternatives in the CSP language:

1. the *external choice*, written -$\Box$-, which forces a process to “follow” whatever the environment offers, and
2. the *internal choice*, written -$\sqcap$-, which imposes on the environment of a process to “follow” the non-deterministic choices made.

The difference becomes more clear if we consider some generalisations of them: Let $\Box x{\in}\{e_1,...,e_n\}{\rightarrow}P(x)$ be an abbreviation for $e_1{\rightarrow}P(e_1)\ \Box\ ...\ \Box\ e_n{\rightarrow}P(e_n)$ (and $\bigsqcap x{\in}A{\rightarrow}P(x)$ analogously). Then the former can be understood as a ”read” and the latter as a ”write” to the process context. This gave rise to the syntactic sugar $c?x{\in}A{\rightarrow}P(x)$ resp. $c!x{\in}A{\rightarrow}P(x)$ for an injective function c into Σ (called *channel*) for reading or non-deterministically writing events along a channel. The failures $\mathcal{F}$ enable to distinguish -$\Box$- and -$\sqcap$- by annotating the traces by the sets of events that a process can *not* engage in. CSP describes the most common communication and synchronization mechanisms with one single language primitive: synchronous communication written $P[\![A]\!]Q$ (P synchronizes with Q over the synchronization set A). With $(c?x{\in}A{\rightarrow}P(x))$ $[\![A]\!]$ $(c!x{\in}A{\rightarrow}P(x))$ the reading process can be set into a process context with a writer since both processes can only proceed if the reader follows the writer. Note that $(e_1{\rightarrow}P \sqcap e_2{\rightarrow}P)$ $[\![\{e_1,e_2\}]\!]$ $(e_1{\rightarrow}Q)$ may result in a deadlock process denoted by *Stop*, and note further, that processes have only to agree on the elements in the $e_1{\rightarrow}(P[\![\{e_1\}]\!]Q) \sqcap e_2{\rightarrow}(P[\![\{e_1\}]\!](e_1{\rightarrow}Q))$. The interleaving

[6] A third component, the divergences $\mathcal{D}$, is not relevant for this paper.

operator $P|||P'$ and the parallel operator $P||P'$ are special instances of the synchronization which result from the empty resp. the universal synchronization set; generalisations of these operators for finite families of processes are written $[\![S]\!] i{\in}A.\ P(i)$ and $|||i{\in}A.\ P(i)$.

The key contribution of CSP is a large set of equivalences (the "laws") and refinement notions based on trace and failures inclusion. In particular, the compositionality of the language is reflected in numerous monotonicity rules which represent a powerful tool for proofs over processes and an alternative to reasoning over coinductive definitions or automata-based constructions.

2.2 Isabelle and Higher-order Logic (HOL)

Isabelle/HOL [7] is a semi-automated proof assistant for higher-order logic (HOL). As an LCF-style theorem prover, it is based on a small logical core to increase the trustworthiness of proofs without requiring — yet supporting — explicit proof objects. Both the logics as well as the kernel-architecture of the implementation are fairly well-studied and attracted over the nearly 40 years of development a fairly large user-community.

Isabelle is available inside a flexible system framework allowing for logically safe ("conservative") extensions, which comprise both theories as well as implementations for code generation, documentation, and specific support for a variety of formal methods. In recent years, a substantial number of theories and system extensions have been collected in the Archive of Formal Proofs [8]. Isabelle/HOL comes with very substantial libaries which have been constructed via conservative extensions; notably *HOL-LCF* capturing Scott's logic of continuous functions [39] and *HOL-Analysis* covering the major part of the mathematics taught in graduate level analysis classes.

2.3 Isabelle and HOL-CSP

Our theory HOL-CSP [42] is based on [45], which has been the most comprehensive attempt to formalize the denotational failure/divergence semantics of [38]. HOL-CSP is a conservative embedding in HOL providing an abstract type α *process* encapsulating the failure/divergence domain parameterized by arbitrary, in particular infinite HOL types α. The type α *process* has been shown to be a Scott complete partial order (cpo) such that processes can be based on HOL-LCF providing a fixpoint theory and continuous function spaces.

Particular effort has been invested in the generalisation of the operators to infinite sets for synchronization and prefixing, which all (with the exception of the hiding operator) have been shown to be continuous wrt. the underlying cpos. As a consequence, HOL-CSP can have events carrying real-time and physical states involving sets of multidimensional vectors, for example.

Example: We define in HOL-CSP notation a process satisfying the recursive equation:

[7] `https://isabelle.in.tum.de/doc/tutorial.pdf`

[8] `https://www.isa-afp.org`

$$P(t) = \mathit{time?}\Delta t{\in}\{\mathit{0}\ ..\ \mathit{5}\} \rightarrow \mathit{calc}!(\mathit{sqrt}(t{+}\Delta t)) \rightarrow P(t{+}\Delta t)$$

where the channels *time* and *calc* are defined via the data-type *event* ≡ *time* $\mathbb{R}$ | *calc* $\mathbb{R}$.

The parameterized process P has the type $\mathbb{R} \Rightarrow$ *event process*. It can engage in an arbitrary real-value from the interval $\{\mathit{0}\ ..\ \mathit{5}\}$ offered in the channel *time*, computes the elapsed time from the initial time and sends via the channel *calc* the square root of the elapsed time (as a rough approximate of a braking curve distance).

In the set of traces $\mathcal{T}(P(\mathit{0}))$, there are, e. g., the scenarios:

- [*time 0*, *calc 0*, *time 1*, *calc 1*, *time 1*, *calc* (*sqrt 2*), ...]
- [*time* π, *calc* (*sqrt* π), *time 0*, *calc* (*sqrt* π), *time 1*, *calc* (*sqrt* (*1*+π)), ...], but also:
- [*time 0*, *calc 0*, *time 0*, *calc 0*, *time 0*, *calc 0*, ...].

Note that the latter scenario, also called Zeno-scenario or time-freeze-scenario, is no particular problem for our framework since the traces monotonically grow and the above process equation has therefore a uniquely defined fixed-point.

Further note that the trace generator of [16] could be combined with our framework. It randomly chooses scenarios out of $\mathcal{T}(P(\mathit{0}))$; depending on the underlying code-generator configuration, more or less precise values for terms such as *sqrt* π can be computed. It is straight-forward to feed this output in conventional visualization tools used in simulator approaches, giving engineers an immediate response when changing our actor models. However, in this paper we focus on techniques that allow to establish properties over *complete* trace sets, not just randomly chosen (large) example-sets which inherently depend on the limitations of calculations.

2.4 Responsibility Sensitive Safety (RSS)

The target of our analysis is described as follows: we aim at finding a suitable model of the safety property “no collision” for the *Responsibility-Sensitive Safety* (RSS), a particular driving strategy that controls acceleration, speed and distance to the car in front. This represents a concrete instance of the aforementioned general modeling framework designed to formally analyse safety properties by proof techniques.

More concretely: we will formalize the concepts of [40] in the framework and formally verify the intended safety property (“no collision” in *all* situations).

1. We outline the RSS as presented in [40]:
    - A formal model and an analysis of the collision hazard
    - Formal definition of a behaviour (the “driving strategy”) compatible with the law (“Duty of Care”)
    - Paper-and-pencil proof that this behaviour ensures global safety (“Utopia is possible”)

2. we provide an instantiation of our general framework for actors as a CSP based model
3. formally verify the reformulation of the problem as an invariant-preservation proof, and
4. analyse extensions of the original paper.

Assumption 1 *RSS makes a number of fundamental assumptions worth being made explicit:*

1. Sensors are perfect*: all actors in the model "know" at certain points in time the absolute physical state (position, speed, acceleration) of all other actors. This implies that no noise on captors like false speed measurements or false LIDAR-results.*
2. Actors are truly autonomous *: when the physical state of all actors is known, the system chooses an acceleration that will be constant in a time interval. No other force, glitch, resistance, etc. will be relevant than this choice.*
3. No actor confusion*: all actors are correctly identified, i. e. there are no scenarios implying confusion of a car with, for example, a pedestrian.*
4. Competent drivers*: all actors follow the same driving strategy.*
5. Topology is respected*: all actors drive on lanes and stay on them.*

The function that chooses the acceleration (dependant on the physical states) is called a driving strategy*, we refer to "classic" RSS as just a particular one.*

For convenience, RSS is introduced in [40] for two cars on a straight lane; these modeling restrictions were stepwise lifted throughout their argument. More precisely, the cars were called *front* and *rear* car. Fig. 1 presents the global scenario.

$$d_{rss} = \left[v_r \ \rho + \frac{1}{2} \ a_{max,accel} \ \rho^2 + \frac{(v_r + \rho \ a_{max,accel})^2}{2 \ a_{min,brake}} - \frac{v_f^2}{2 \ a_{max,brake}} \right]_+$$

Assumption 2 *Furthermore, the RSS assumes the following modeling parameters:*

- *reaction time called ϱ of all actors, assuming to be the overall time interval comprising capturing time, processing time, time lapses for communication and the reaction time of actors.*
- *minimal longitudinal distance that actors have to respect is called d_{rss}.*
- *speeds for "rear" and "front" cars: v_r,v_f*
- *the model assumes three accelerations, and they are assumed to be equal for the front and the rear car: $a_{max,brake}$ is the maximal negative acceleration of both cars, $a_{min,brake}$ is the minimal negative acceleration of the rear car when braking is required and $a_{max,accel}$ the maximal positive acceleration.*

This results in the calculation of the minimal distance d_{rss} as defined in the formula above and visualized in Fig. 1.

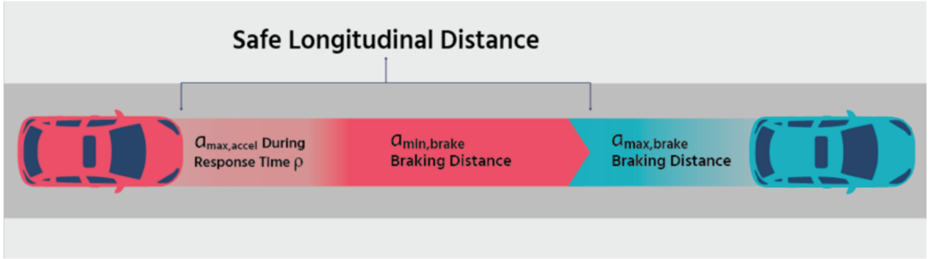

Safe Longitudinal Distance
$a_{max,accel}$ During Response Time ρ
$a_{min,brake}$ Braking Distance
$a_{max,brake}$ Braking Distance

Fig. 1: Safe Longitudinal Distance in RSS

3 Our CPS Modeling Framework and RSS Instances

3.1 Foundations: Actors as Processes

Our view that *actors* and *scenarios* in AV simulations are *processes* in the sense of CSP lends itself to an extremely compact notation, and a proof based approach to the analysis of large classes of functional and abstract scenarios.

In the following, we will illustrate how the CSP formula for scenarios provides a formal semantics that captures user defined scenarios at their different levels of abstraction, as well as their subsumption relationships through CSP refinement.

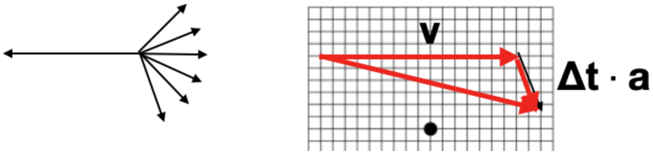


Fig. 2: Acceleration Vector Space vs. Speed Vectors

In particular, the Assumption 1 can be altogether captured by a hypothetical process[9] that:

1. chooses non-deterministically a Δt and communicates this to its environment (this is the time interval in which the actors will be observed in their physical state)
2. ... and collects all individual states of actors to the global state $\sigma_g \in \Sigma$ of the scene on which the actors all have to agree via synchronization:

$$demon \equiv \bigsqcap \Delta t \in \mathbb{R}_+ \rightarrow \Box \sigma_g \in \Sigma \rightarrow demon$$

[9] We call this global process “Maxwell demon” after a thought experiment by the physicist James Maxwell in the context of thermodynamics; in our setting, the demon “knows” all individual states of all actors at his choice of the time Δt.

The global schema of an actor, which is a parameterised process depending on an *actor identity id* and a *driving strategy ds*, is then defined as a process that "reads" the time interval Δt set by the demon, and calculates on the basis of the prior state σ_g and Δt via the driving strategy *ds* this set of possible accelerations for *id*. The *kinematics* maps the local state of the actor pointwise into the set of possible accelerations, which gives a set of future local states. The actor chooses non-deterministically a global state where "his" substate must agree with one of these. Note that we usually bound the time-space sometimes by some model parameter ϱ.

More formally, this is captured in HOL-CSP as follows:

$$actor_{id}\ ds\ \sigma_g \equiv \Box\ \Delta t{\in}\mathbb{R}_+ \rightarrow \sqcap\ \sigma'_g \in \{\Sigma \mid \Sigma[id] \in moves\} \rightarrow actor_{id}\ ds\ \sigma'_g$$

where $moves \equiv (kinematics\ (\sigma_g[id])\ \Delta t)$ ' $(ds\ id\ \sigma_g)$.

Note that the global states Σ are just a family of local states; $\Sigma[id]$ therefore denotes the local state of actor *id*. The function *moves* abbreviates the calculation of the driving strategy on a global state σ_g, which gives a set of accelerations for the actor *id*, and its translation into the set of resulting physical states of the actor via the *kinematics* of the actor.

Note that our concept of *actor* is fairly distinct from other modeling approaches in the CPS domain attempting to determinise them [12]. Rather than assuming a set of functions for positions and define their derivatives speed and acceleration by postulating some constraints on them, we use *non-determinism* to model the set of possible trajectories and time intervals under consideration (see Fig. 2). In particular, actor decisions may result in non-continuity and non-differentiality of observables.

Since we consider *all* time intervals and *all* possible accelerations of an actor, these two points of view can be made mathematically equivalent. However, having actors as explicit processes paves the way to a proper treatment of refinement and the specific problems due to discretisation (sampling) (see Fig. 4). Moreover, we argue that our point of view lends itself more easily to driving strategies that take the communication with the environment into account, be it by explicit modeling of interaction with the signaling infrastructure or be it by explicit modeling of exchange protocols between cars as shown below.

On the basis of the *demon* and the *actors*, autonomous car simulations can be defined as composition of the following processes (see schema Fig. 3). The *demon* is synchronized via the disjoint set of times and global states with a family of actor threads, that live their life independently from each other except in the synchronization points set by the demon in Fig. 3, and their synchronizations by agreement on a global state.

$$S\ \sigma_0 \equiv demon\ [\![\mathbb{R} \uplus \Sigma]\!]\ (|||\ id \in IDS.\ actor_{id}\ ds\ \sigma_0)$$

The key notion of *scenarios* in the autonomous car domain is then easily defined as the traces of this process: $\mathcal{T}(S\ \sigma_0)$ where σ_0 is the initial global state (the initial *scene*).

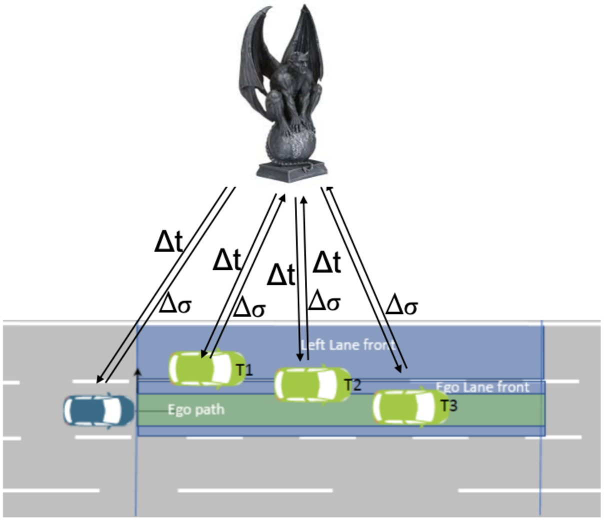


Fig. 3: Maxwell's demon vs. Cut-In Scenarios

Note that the above definition of the scenario process S does not provide any interaction between the actors; this is sufficient for our main application, aimed at verifying the RSS driving strategy. In general, if we wish to allow communication *between* the actors (e. g., a car reads from a traffic light directly its status, or, a car communicates "driving intentions" to another one for optimization purposes), we will use the generalized form:

$$S_P{}' \; \sigma_0 \equiv \mathit{demon} \; [\![\mathbb{R} \uplus \Sigma]\!] \; (([\![\; I \;]\!] \; id \in \mathit{IDS}. \; \mathit{actor}_{id} \; ds \; \sigma_0) \setminus \{I\})$$

where I is the set of *internal* events used only for the communication between actors, and - \ - the hiding operator of CSP; this concealment allows actors to pursue local communication without changing the overall process composition and impacting proofs over the general architecture.

The assumptions *truly autonomous* and *constant acceleration* in Δt lead to a fairly conventional kinematics for our analysis of RSS. If we assume a local physical state of an actor to consist of the triple of the real-vector functions position x, speed v and acceleration a, then the future state $(x', \; v', \; a')$ can be computed based on the constant acceleration a_0 by:

$$\begin{aligned} x' &= x + \Delta t * v + (\Delta t^2/2) * a_0, \\ v' &= v + \Delta t * a_0, \\ a' &= a_0 \end{aligned}$$

However, our framework is not restricted to this kinematics; since HOL-Analysis provides the theory for the derivation and integration operators, *deriv* and *integrate*, it is perfectly possible to model the kinematics as the solution of a differential equation system:

$$
\begin{aligned}
\mathit{SOME}\ (x',v',a').\ \ & v' = \mathit{deriv}\ x\ \wedge\ a' = \mathit{deriv}\ v \\
& \wedge\ (x',v',a') = M\ (x',\ v',\ a') \\
& \wedge\ x'(\mathit{0}) = x(\mathit{0})\ \wedge\ v'(\mathit{0}) = v(\mathit{0})\ \wedge\ a'(\mathit{0}) = a_0
\end{aligned}
$$

Here, the matrix of higher-order functions M is a placeholder for arbitrary (linear) combinations of (derivatives) of x', v', a'. The HOL Hilbert-Choice *SOME* yields the solution of this ODE system, provided that it is uniquely defined by M. Even in this case, however, note the existence of these functions does not mean that we have in general an calculative means to construct them. Isabelle/HOL just provides a means to model ODEs and to verify a given solution by formal proof, at least in principle. [10]

So far, we did not make any restriction on the type of elements in our time space, so it can be just the set of real numbers $\mathbb{R}$. In our concrete model instances, we will usually require that time is positive, the interval Δt can be infinitesimally small and is in any case assumed to be substantially smaller than a certain model parameter, the reaction time ϱ.

The view "actors are processes" also lends itself to a straightforward notion to compare driving strategies as a trace refinement problem: given two driving strategies ds_1 and ds_2, we can state for any initial global scene σ_0 that:

$$\forall\, \sigma_0.\ \mathit{actor}_{car}\ ds_1\ \sigma_0 \sqsubseteq_{\mathcal{T}} \mathit{actor}_{car}\ ds_2\ \sigma_0$$

which is to say that the sets of possible scenarios are included one in another:

$$\forall\, \sigma_0.\ \mathcal{T}(\mathit{actor}_{car}\ ds_2\ \sigma_0) \subseteq \mathcal{T}(\mathit{actor}_{car}\ ds_1\ \sigma_0)$$

Note that the (trace)-refinement notion $\sqsubseteq_{\mathcal{T}}$ in CSP is defined contravariant wrt. the subset inclusion, since the traditional view on process refinement is P refines P' iff it is more deterministic and more defined [38].

3.2 An Extensible Model of Scenes

So far, there was no need to detail the structure of the state space of scenarios S, and the concrete global scenes such as the initial Σ_0 in particular.

This choice of abstraction level is deliberate: the relevant ISO standard SOTIF (ISO 21448) [44] for road vehicles introduces a classification of scenarios in several categories called

> [...] *known not hazardous*, *known hazardous*, *unknown not hazardous* and *unknown hazardous*, depending on whether the scenario is known during the design of the system or discovered during the test phase, and depending on whether the scenario does not destabilize the system or cause it to fail.

[10] The precise formalisation of generalized kinematics in HOL can be found in the last section of the theory `Framework.thy`.

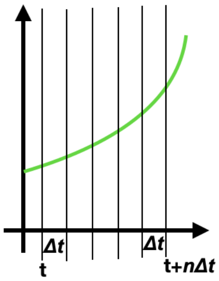


Fig. 4: Samplings of Continuous Observables

In our work, it is also a question of defining an approach to list the cases covered and compliant (with success criteria) by proof or simulation, in order to be able to identify a safe perimeter of use (Operational Design Domain or ODD). In the spirit of the SOTIF standard, this amounts to (see [44], p. 20):

– *Perform a risk acceptance evaluation of [known hazardous scenarios] based on the analysis of the intended functionality.*
– *Reduce the probability of known scenarios causing hazardous behaviour [...] to an acceptable level of risk.*
– *Reduce the probability of the unknown scenarios causing potentially hazardous behaviour [...] to an acceptable level of risk.*

We do not propose a truly probabilistic setting for our analysis, since the probabilistic weights on, e. g., Markov automata, can only be validated on the basis of a very large real-world experimental dataset. Such experimental data will only be available in a very late stage of the development. While remaining in a strictly *possibilistic* setting, a refinement notion in the sense "more deterministic, more defined" as proposed by CSP captures the essence of the desire to avoid "known hazardous" scenarios while maintaining the prognostic power of models and their analysis.

Abstraction resulting in non-determinism and its analytic control via CSP refinement (rather than probability) is therefore one answer to the objective of the SOTIF standard to master *known hazardous* and even *unknown hazardous* risks of autonomous vehicles.

Parameterised specifications and extensible state-spaces as offered by HOL-CSP will be another answer. In the following, we address the problem to specify the internal structure of actors in an extensible way. As we will see, extensibility of our models will even allow to establish properties which remain valid under extensions of the model, as long as they respect a certain monotonicity expressed via typing constraints instances of the underlying framework.

We will formalize the basic *actor state* via a specification construct in Isabelle/HOL called **record** inspired by many programming languages.

```
record ('υ::real-normed-vector) as = — for actor state
  pos   :: 'υ                    — current position
  speed :: 'υ                    — current speed
  acc   :: 'υ                    — current acceleration
```

Actor states are explicitly parameterized by the type variable $'\upsilon$ which is constrained by the type-class *real-normed-vector* imported from the HOL library. This class provides a theory for vector spaces such as,e. g., $\mathbb{R}^n$ or $\mathbb{C}^n$ that possess a scalar product in $\mathbb{R}$.

The record notation generates a theory based on the semantics of cartesian products $\tau_1 \times \ldots \times \tau_n \times '\alpha$ where the τ_i correspond to the types of the attributes (in our case: $'\upsilon$) and the type-variable $'\alpha$ stands for an "extension field" of the record. For the actor states, this type is given the alternative type notation $('\upsilon,\ '\alpha)$ *as-scheme* which we will denote $('\upsilon,\ '\alpha)$ *as-* for short. The *attributes* of a record become projection functions into the cartesian tuple. The constructors of a record can be denoted by $(\!|\ pos = a,\ speed = b,\ acc = c,\ \ldots = m\ |\!)$, and the resulting projection rules like $pos\ (\!|\ pos = a,\ speed = b,\ acc = c,\ \ldots = m\ |\!) = a$ were derived from the underlying semantics of the cartesian products automatically. Note that a property P established on some expression $e{::}('\upsilon,'\alpha)$ *as-* remains valid for any type instance of $'\upsilon$ or $'\alpha$, so in particular $e{::}('\upsilon,unit)$ *as-* or $e{::}('\upsilon,some\ extension)$ *as-*; this is a fundamental property of HOL. Now we can extend the actor states with some more structure, for example by saying that it has a physical extension (relevant for defining collisions) and a specific set of possible accelerations of an actor. In this model, this set of accelerations can also take braking and nonlinear movement into account:

```
record ('υ::real-normed-vector) as_range = ‹'υ as› +
  acc-range       :: ‹'υ set›
  extension-field :: ‹'υ set›
```

This Isabelle/HOL syntax generates an *extension* of the $'\upsilon$ *as-* type. technically, this means that a new type $('\upsilon,'\alpha)\ as_{range}$- is defined as synonym to

$$('\upsilon,(\!|\ acc\text{-}range :: '\upsilon\ set,\ extension\text{-}field :: '\upsilon\ set,\ \ldots :: '\alpha\ |\!))\ as\text{-}$$

and again, that properties proven over objects of type $('\upsilon,'\alpha)\ as\_$ will remain valid for $('\upsilon,'\alpha)\ as_{range}$-, whatever the future extensions $'\alpha$ and $'\upsilon$ are.

The patient reader might ask why are we explaining this in this gory level of detail. The point is that our formal model is *extensible* and therefore able to make predictive statements over *some form of unknown hazardous* behaviour, notably over actors based on actor states that were not yet modeled in the subclass hierarchy.

Such subclass hierarchies are used in common simulator technology in the autonomous car domain for the modeling of so-called "ontologies" for the actor states in, e. g., the MOSAR platform [11] or ASAM's OpenSCENARIO 2.0 [12]. An extract of MOSAR data modeling is shown in Fig. 5.

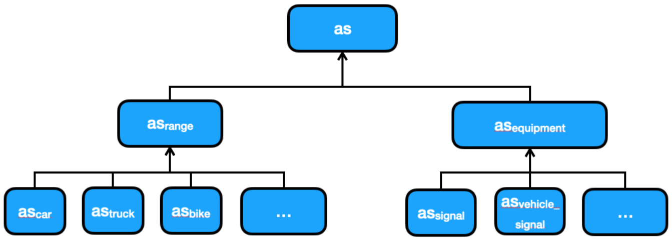


Fig. 5: Extract of a MOSAR Ontology

These kinds of models can be, if suitably extended by typed information such as the vector spaces, compiled in a straightforward way into our framework.

To wrap up, it remains to clarify the state space of global scenes S as a map from the set IDS to the root class of our actor state spaces:

type-synonym $('\upsilon,'\alpha)$ *scene-* $=$ ‹$id_{actor} \Rightarrow ('\upsilon,'\alpha)$ *as-*›

... where IDS is just the set of all elements of the type id_{actor}.

4 Formal Analysis by Machine-checked Proofs

To be very clear, in our view, the work presented in [40] represents a milestone towards a clarification of basic concepts in the autonomous car domain, and a major step towards their formal analysis. As the regulatory standards such as the SOTIF [44] seem still to be emerging, works like these play an imminent role in clarifying the foundations. A rigorous application of state-of-the-art techniques and tools represent a rewarding target in order to increase the trustworthiness of this complex and safety critical technology. We believe there is relevance for all three: the scientific community, the industrial stakeholders as well as regulators.

4.1 A Critique of the Paper-and-Pencil Proof of "classic RSS"

Unfortunately, the paper-and-pencil proofs presented in [40] revealed a number of substantial shortcomings. Discovering these is not unusual when doing formal, machine-assisted proof by an interactive proof assistant, be it in traditional mathematical textbooks or scientific papers in theoretical computer science. Paradoxically, it is also not unusual that the presented proofs are incorrect,

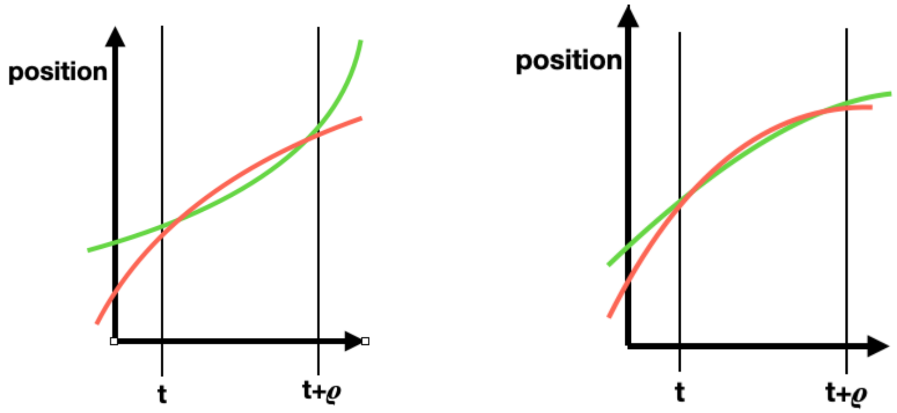
[11] `https://www.mosar.io`

[12] `https://asam-ev.github.io/public_release_candidate/asam-openscenario/2.0.0/welcome.html`

but due to ingenious human intuition, the overall conjecture is true nevertheless. Often, this means that the theorem has to be established by a different proof argument or technique.

This is also the case for RSS, where the key conjecture is circumscribed by:

> *if the fundamental assumptions listed in Sect. 2.4 are met, RSS is safe in the sense that there will be no collision.*



(a) Front accelerating, back braking.

(b) Front braking, back braking.

Fig. 6: Erroneous Arguments in RSS Safety Proof

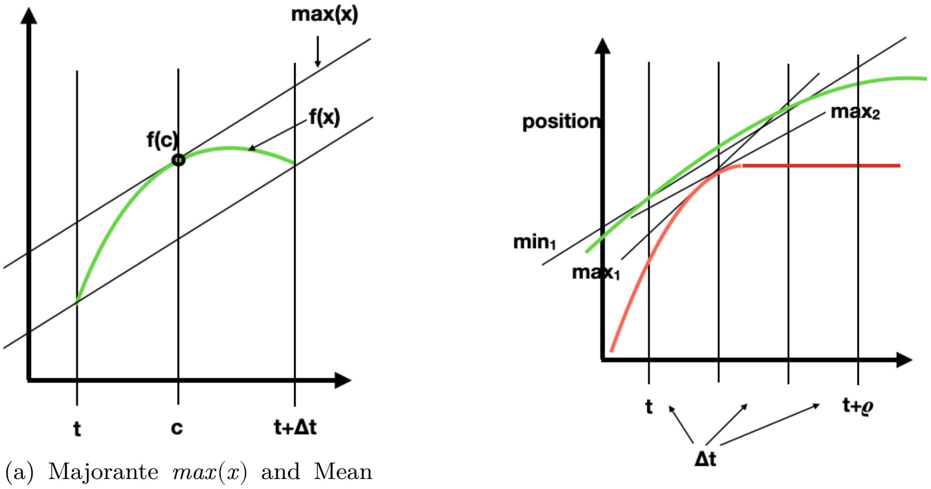
The proof uses arguments roughly as follows for any time interval defined by the reaction time ϱ of the car:

- if the front car is in front at the beginning and the end of the time interval, and if the front car is accelerating and the back car is braking, there will be no collision.
- if the front car is in front at the beginning and the end of the time interval, and if both cars are braking, there will be no collision.

Fig. 6 shows for both cases that the argument does not hold: there are monotonous acceleration and braking functions that actually intersect under these premises. A particular difficulty is that the derivative of the braking function is non-continuous: if the car stops, its position becomes the constant function rather than the square root. The authors of [40] mention this problem without actually addressing it in their proof.

Conceptually, it is unsatisfactory in the proof not to distinguish the reaction time ϱ from the sampling interval Δt. The former is a system parameter, it is an objective of the safety target to establish that it is chosen appropriately in the concrete system design to establish safety. This has to be shown for any sufficiently small Δt, which is part of the safety property, not a system parameter. Identifying these results in an improper separation between analysis and system-design to be analysed.

In our model, there is sufficient freedom to establish that within a chosen Δt, and a given set of assumptions, some safety property holds, and this will be true for all smaller Δt.



(a) Majorante $max(x)$ and Mean Value t.

(b) Section-wise Approximations.

Fig. 7: Calculation via Majorants and Minorants.

For example, one can apply the mean value theorem to construct a majorant $max(x)$(or minorant, respectively) in the interval $[t..t{+}\Delta t]$ (see Fig. 7 (left)). In our concrete problem, we suggest a section-wise argument based on majorants and minorants for the position functions within Δt, which is supposed to be smaller than ϱ (see Fig. 7 (right)). Inside a proof, discontinuities will have to be discussed systematically within all intervals Δt.

4.2 The Target: “Classic” RSS and three Variants

We instantiated our framework to four cases:

- RSS in the sense of the paper: two cars, the front car f and the back car b, in a linear lane. The situation covered and analysed in [40].
- RSS^n — an extension to scenarios with n cars alltogether respecting RSS, thus replacing a mere “proof-by-analogy” argument in [40].

- RSS$^+$ — an optimized variant of RSS allowing for shorter safety distance and better comfort,
- and finally the RSS^{lateral}-case were we consider the lateral distance to cars and obstacles.

For all cases, we provide a formal proof of the *safety*: if the initial scene does not contain a collision – any two cars have different positions – then in all *scenarios*, so all $tr \in \mathcal{T}(S_P\ \Sigma_0)$, there will be no collision.

The proofs for the RSS scenarios boil immediately down to an induction proof over all *tr* that start with an initial scene Σ_0 that is safe, and the induction step establishes preservation. In more detail, the induction proof assumes for any time interval $\Delta t < \varrho$ the transition from a safe scene will result in a safe scene after Δt.

4.3 Formal Safety Proof of "classic" RSS

We adopt the most general and non-deterministic definition of the RSS driving strategy to handle all cases:

$$\begin{aligned} drive_{rss}\ \ldots \equiv\ &\mathit{if}\ d_{rss} \leq d_{real}\ \mathit{then}\ [-a_{max,brake}\ \cdots\ a_{max,accel}] \\ &\qquad\qquad\qquad\quad \mathit{else}\ [-a_{max,brake}\ \cdots\ -a_{min,brake}] \end{aligned}$$

If the situation is safe, $d_{rss} \leq d_{real}$ meaning that the real distance is greater than the RSS safety distance, we can apply any acceleration in the interval $[-a_{max,brake}\ \cdots\ a_{max,accel}]$ (from max braking to max acceleration), but if the situation is not safe, the car should brake with any value within $[-a_{max,brake}\ \cdots\ -a_{min,brake}]$.

In order for this induction to be provable, we need to reinforce it with an invariant that is stronger than the non-collision predicate alone. For this purpose we define a new distance d_{min} that is shorter than d_{rss} as follows:

$$cd_{min} = [b_r - b_f]_+ = [{v_r}^2/2a_{min,brake} - {v_f}^2/2a_{max,brake}]_+$$

where

$$[x{::}int]_+ \equiv max(0,\ x)$$

Compared to the RSS safety distance d_{rss}, the distance d_{min} only considers the difference in braking distances (between rear and front cars) and does not take into account a possible acceleration during the reaction time ϱ.

The induction invariant is $d_{min} < d_{real}$. It states that the driving strategy keeps the real distance (the difference between the front and the rear car positions) greater than d_{min} in all circumstances. Especially during a non-safe period where $d_{real} < d_{rss}$ when the real distance is under the safety distance as defined in RSS and the rear car applying the $drive_{rss}$ strategy is supposed to be braking.

The proof of the induction invariant requires many smaller steps and distinguishing many cases. Let us first introduce many variables/measures:

- p_f, v_f, a_f and $b_f \equiv {v_f}^2/2a_{max,brake}$ are respectively the position, the speed, the acceleration and the braking distance characterizing the state of the front car right before Δt,
- p'_f, v'_f, a'_f and $b'_f \equiv {v'_f}^2/2a_{max,brake}$ characterize the state of the the front car right after Δt,
- p_r, v_r, a_r, $b_r \equiv {v_r}^2/2a_{min,brake}$, p'_r, v'_r, a'_r and $b'_r \equiv {v'_r}^2/2a_{min,brake}$ characterize similarly the two states of the rear car.

Then the invariant proof consists of:

- showing that when the rear car applies the *drive*$_{rss}$ driving strategy, we preserve:
$$\begin{aligned} &d_{min} = [b_r - b_f]_+ < p_f - p_r = d_{real} \\ &\qquad \Longrightarrow d'_{min} = [b'_r - b'_f]_+ < p'_f - p'_r = d'_{real}, \end{aligned}$$
- ... and concluding safety as a direct consequence of the invariant statement as d_{min} is positive by definition, we deduce that d_{real} will remain strictly positive.

The invariant proof requires 700 lines of Isabelle/Isar proof code using automated proof procedures. Case distinctions lead to subcases representing intermediate results. Due to lack of space, we will only provide four of them: [13]

- $[b_f - b'_f]_+ \leq p'_f - p_f$: The front car will travel a distance greater than the difference between its braking distances before and after Δt. This is due to its acceleration a_f which is necessarily weaker than the max brake $-a_{max,brake}$.
- $d_{real} < d_{rss} \Longrightarrow p'_r - p_r \leq b_r - b'_r$: If the rear car applying the RSS driving strategy is in a state where the safety distance is not respected, it is therefore in a braking phase where its acceleration a_r belongs to the interval $[-a_{max,brake} \cdots -a_{min,brake}]$. So, it will travel a distance shorter than the difference between its braking distances before and after Δt.
- $d_{rss} \leq d_{real} \Longrightarrow b_r - b_f < p_f - p'_r$: In a safe state where the RSS safety distance is respected, we prove a very strong statement far from being trivial: the rear car will move forward without even reaching the previous position of the front car, i.e. even if the front car does not move forward, the vehicle will not violate d_{min} during that step.

Note that the RSS safety distance formula assures that the computed distance has a small value and may even be zero when the front car has a higher braking distance $b_f \gg b_r$, i.e. we can drive very close behind a car with weak brakes! As consequence the previous statement inequality only helps to prove non-collision in the case where $b_f \leq b_r$, otherwise we proved a new inequality:

$$b_r < b_f \wedge d_{rss} \leq d_{real} \Longrightarrow p'_r - p_r - b_r \leq p'_f - p_f - b_f$$

In a safe state where the RSS safety distance is respected and the rear car has a shorter braking distance (implying that $d_{rss} \approx \mathit{0}$ is close to 0), we succeed to prove that the difference between the travel distance and the braking distance of the rear car is shorter.

[13] The reader may explore the complete Isabelle proof in [11].

4.4 Formal Safety Proof of RSS for n Cars

The original paper claims that the extension to an n Car-scenario, where all cars respect the RSS driving strategy, would be straight-forward. In this case, we agree with the authors: the proof just adds another induction layer with about 100 lines of proof code to arrive at the safety theorem. The model and verification technique has strong similarities with the "Platoon Control Strategies" analysed in [37] with the HOL-light system.

4.5 Formal Proof of "optimized" RSS$^+$

The RSS strategy is based on the worst case assumption wrt. the safety distance d_{rss} that the rear car would apply a maximum acceleration $a_{max,accel}$ during the entire response time ϱ. This is too strong an assumption since we know our current acceleration which can be considerably smaller $a_r \ll a_{max,accel}$. In our optimized version of RSS that we call RSS$^+$, we adopt a shorter safety distance by replacing occurrences of $a_{max,accel}$ in the d_{rss} formula by a_r.

$$d_{safe}\ a_r = [\varrho\ v_r + (\varrho^2/2)\ a_r + (v_r + \varrho\ a_r)^2/2a_{min,brake} - {v_f}^2/2a_{max,brake}]_+$$

The RSS safety distance is then a particular case when the maximum acceleration is applied: $d_{rss} = d_{safe}\ a_{max,accel} \gg d_{safe}\ a_r$. We managed to prove that this shorter distance is safe.

As a consequence, we can notably improve the RSS driving strategy while adding more comfort, i. e. by ruling out short-term brake-and-accelerate fluttering. As recalled in the previous section, the RSS driving strategy was binary: we either drive with any acceleration under $a_{max,accel}$ when safe $d_{rss} \leq d_{real}$ or we brake otherwise. Thanks to our optimized safety distance taking into account a_r, the advantage is twofold:

1. we can avoid braking in many cases where $d_{safe}\ a_r \leq d_{real} < d_{rss}$ while keeping safe.
2. we can adapt our acceleration to keep $d_{safe}\ a_r \leq d_{real}$ without braking $a_r \geq -a_{min,brake}$.

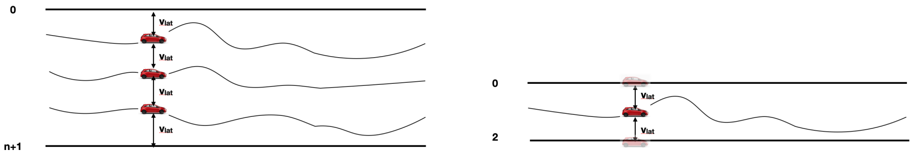
We define our "smoothened" version of the RSS driving strategy as follows:

$$\begin{aligned} drive_{smooth}\ \cdots \equiv\ & \{-a_{max,brake}\ \cdots\ a_{max,accel}\} \\ & \cap \{a.\ a > -a_{min,brake} \longrightarrow d_{safe}\ a \leq d_{real}\} \end{aligned}$$

We had been able to profit from the previous RSS proof structure: the RSS$^+$ proof is more complex but the splitting of cases remains basically unchanged. Overall, the safety proof is about 200 loc longer.

4.6 Model and Proof of the RSS^{lateral} - Strategy

In the previous sections we considered driving strategies that control the longitudinal distance to other actors. In this section, we treat the *lateral* behaviour – for short, the second dimension relevant for AV movements. In the original paper, it was claimed that an extension to the lateral case would be straightforward, without presenting any details neither for the generalized strategy nor the invariant.



(a) The Lateral Scenario. (b) Consequence: A simple Lane-Model.

Fig. 8: Lateral Driving Strategies

In this section we present such a generalization and its safety proof [14]. We consider a scenario for N cars as shown in Fig. 8 (left). The sweetspot of this model is that also gives a driving strategy for staying on a linear lane: by adding two actors which just represent the borders as shown in Fig. 8 (right).

We found the generalization to be very complex, mostly due to the fact that RSS only considers positive speeds: if the front car gets too close, then the rear car just stops. In the lateral case, however, negative relative speeds naturally arise which have to be covered.

In more detail, we need to define the auxilliary functions:

1. $a_{maxaccel}$::*id*$_{actor}$ ⇒ *real* and $a_{minaccel}$:: *id*$_{actor}$ ⇒ *real*: the maximum acceleration in direction of the right and of the left of the car ($a_{minaccel}$ is negative).
2. d_{real} :: *real scene* ⇒ *id*$_{actor}$ ⇒ *real*: actual distance between a vehicle and its right neighbor (the distance with its left neighbor is given by d_{real} for the car with *id*$_a$ − *1*.
3. d_{rss}:: *real scene* ⇒ *id*$_{actor}$ ⇒ *real*: Safe lateral distance (to the right, still).
4. d_{min}:: *real* ⇒ *real* ⇒ *id*$_{actor}$ ⇒ *real*: Minimal safe distance corresponding to immediate response $\varrho = 0$ - the invariant of the proof being that d_{min} is always respected.
5. *drive*$_{safe}$ *id*$_a$ δt σ_g : the function computing the set of possible lateral accelerations, i. e. the core of RSS^{lateral}.

[14] The details of the Isabelle formalization can be found in the theory `RSS_Ncars_lateral`

For reasons of limited space, we will only show the definition of the latter:

```
$\{$if speed $(\sigma_g\ id_a) \leq 0 \wedge d_{rss}\ \sigma_g\ (id_a{-}1) > d_{real}\ \sigma_g\ (id_a{-}1)$
 then if speed $(\sigma_g\ id_a) = 0$
      then $0$
      else $a_{minaccel}\ id_a$
 else $-a_{maxaccel}\ id_a$
 ..
 if speed $(\sigma_g\ id_a) \geq 0 \wedge d_{rss}\ \sigma_g\ id_a > d_{real}\ \sigma_g\ id_a$
 then if speed $(\sigma_g\ id_a) = 0$
      then $0$
      else $-a_{minaccel}\ id_a$
 else $a_{maxaccel}\ id_a\ \}$
```

The safety proof is the most complex one in this paper. It requires 64 case distinctions, nearly all involving non-linear approximations and majorant/minorant reasoning as before.

5 An Application to Safety Tests of Autonomous Vehicles

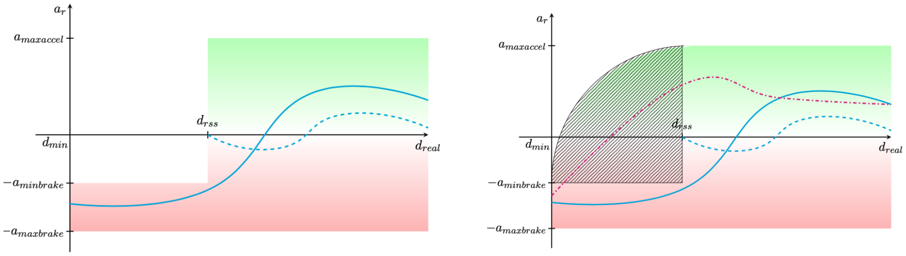
5.1 Safety Validation in the Context of a $V\&V$ Process

The Verification and Validation of AVs relies on evaluations during actual or simulated road tests, described under the form of scenarios (see [44,46]). Scenarios were divided further into classes using decreasing levels of abstraction: abstract, logical, concrete (see [2]) in order to cope with the exploding trace-spaces. Instead of simulations, so finite and usually very large sets of concrete, logical and abstract scenarios, we have shown how semi-automated formal proof can *guarantee exhaustive coverage* for infinite and uncountable scenario classes.

Nevertheless, the approach demonstrated refers to a *model* and only indirectly to reality. In this section, we will show how proofs can be used to produce a senseful classification for actual road tests, a.k.a system integration tests, which validate both our *model and its underlying assumptions* as well as its *implementation* within a concrete AV system.

In a system integration test, the vehicle is considered as a black box, embedded in an omniscience of the observer-validator, which makes the hypothesis of perfect sensors much less problematic. This hypothesis can be attained in a test environment providing proper instrumentation during real-world road tests.

The RSS driving strategy, or "proper response" as named and described in [40], reinterpreted from the point of view of Validation, provides us with a *sufficient condition* for safety with regard to the collision hazard. In other words, any vehicle whose *observed* acceleration a satisfies the rss_{motion} formula given in Sect. 4 at any time is sure to be safe, i. e. any vehicle whose driving strategy lies within the envelope defined by the RSS is sure to enforce this invariant, as illustrated in Fig. **??**.



(a) Driving Strategy RSS.

(b) Driving Strategy RSS$^+$.

Fig. 9: Acceleration Trajectories of Two Strategies ($v_r = v_f > 0$).

5.2 From Proofs to Functional Tests

We suggest to use the proof structure to construct a finite set of abstract test classes. The approach is roughly similar to the decomposition of a specification into its Disjunctive Normal Form and a selection of a test set to cover its clauses, a method going back to [13] and used in many disguises in functional testing.

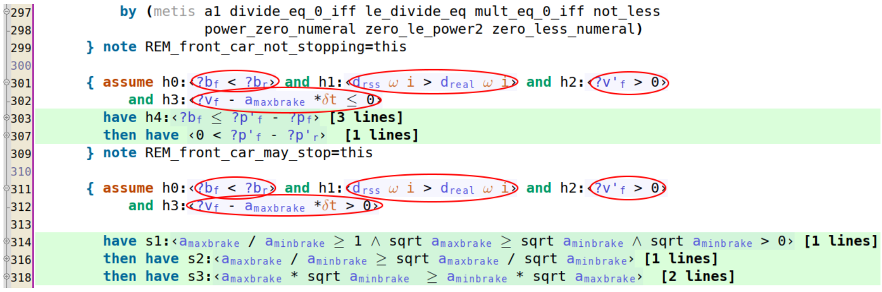
The proof structure inside the induction step consists of a number case distinctions which represent "tipping points" within continuous behaviour. Since these case distinctions usually reflect that between such tipping points, one proof argument works for all instances, it can be speculated that an implementation of this function might work in this interval uniformly, and is therefore a noteworthy target to test. In any case, if the proof needed a different argument to establish the safety property, it is relevant to see if the implementation behaves well, in particular if test values close to the tipping points were chosen.

For example, following the proof, the validation activity can be broken down to abstract test classes such as:

- whenever $d_{real}{<}d_{rss}$, the observed acceleration of the automated vehicle must be below $-a_{min,brake}$;
- otherwise, the observed acceleration of the automated vehicle must be comprised between $-a_{max,brake}$ and $a_{max,accel}$. In other words, there is nothing to check, since these are the physical limitations of the vehicle.

Using the structure of the proof gives us the guarantee that the partitioning into abstract test cases is complete. In Fig. 10, we illustrate how to "manually" extract these test cases from the structure of the Isabelle/Isar proof: for example, the circled sub-conditions give rise respectively to abstract scenarios 3 and 4 in Table 1. For driving strategy $drive_{rss}$, the analysis of the proof tree in order to derive test cases is the following:

- subcases verifying $d_{rss}{\leq}d_{real}$ do not generate a test: if the initial situation is safe, the rear vehicle can have any acceleration within $[-a_{max,brake}, a_{max,accel}]$ during ϱ;



```
          by (metis a1 divide_eq_0_iff le_divide_eq mult_eq_0_iff not_less
                    power_zero_numeral zero_le_power2 zero_less_numeral)
      } note REM_front_car_not_stopping=this

      { assume h0:‹?b_f < ?b_r› and h1:‹d_rss ω i > d_real ω i› and h2:‹?v'_f > 0›
           and h3:‹?v_f - a_maxbrake *δt ≤ 0›
        have h4:‹?b_f ≤ ?p'_f - ?p_f› [3 lines]
        then have ‹0 < ?p'_f - ?p'_r›  [1 lines]
      } note REM_front_car_may_stop=this

      { assume h0:‹?b_f < ?b_r› and h1:‹d_rss ω i > d_real ω i› and h2:‹?v'_f > 0›
           and h3:‹?v_f - a_maxbrake *δt > 0›

        have s1:‹a_maxbrake / a_minbrake ≥ 1 ∧ sqrt a_maxbrake ≥ sqrt a_minbrake ∧ sqrt a_minbrake > 0› [1 lines]
        then have s2:‹a_maxbrake / a_minbrake ≥ sqrt a_maxbrake / sqrt a_minbrake› [1 lines]
        then have s3:‹a_maxbrake * sqrt a_minbrake  ≥ a_minbrake * sqrt a_maxbrake›  [2 lines]
```

Fig. 10: Sub-conditions Establishing Abstract Test Classes ("Logical Scenarios")

- subcases verifying $d_{real}<d_{min}$ also do not generate a test: they correspond to proofs by contradiction and can be interpreted as infeasible scenarios of the AV where a collision is inevitable;
- subcases verifying $d_{min}\leq d_{real}<d_{rss}$ generate test classes: if the initial situation is hazardous, the rear vehicle must brake, by having an acceleration below $-a_{min,brake}$.

The latter subcases divide into two main categories, generating each two test classes in Table 1:

1. $b_r - b_f$ is positive at $t_0+\varrho$ (tests 1 and 2);
2. $b_r - b_f$ turns from positive at t_0 to negative at $t_0+\varrho$ (tests 3 and 4).

The term $b_r - b_f$ represents the difference between the two vehicles worst case braking distances: when positive, the rear car braking distance is potentially longer than that of the front car. Note that we did this extraction of abstract test clauses by hand at present, but this process is entirely syntactic and could be automated.

As usual in functional testing, a small number of abstract test-cases seems not to be suited for a system test; in particular, if the usual uniformity hypothesis ("if the test passes from one pick of the class, we assume correctness for the entire class") is applied. Combinations with a borderline analysis are usually applied to get a finer set of concrete tests.

We propose therefore another test selection strategy, which is described by "select a uniform grid out of the test class". We observe that the space of initial conditions can be reduced to only three depending dimensions: v_r, v_f and d_{real}. Hence, it can be represented as in Fig. 11a, where the green surface corresponds to $d_{real} = d_{rss}$ and the red surface corresponds to $d_{real} = d_{min}$.

This shapes our Design of Experiments (DOE): as explained above, the subspace located "before" the green surface represents situations where the AV may choose any acceleration; the subspace located "after" the red surface represents situations of potential inevitable collision, unreachable under the assumption that all AVs respect the RSS driving strategy. The grid represented on Fig. 11b

No	Test Case description		Conditions
1	t_0,	non-safe distance	$d_{rss} > d_{real}$
		minimal distance	$d_{min} \leq d_{real}$
	$t_0+\varrho$,	r-car has a longer braking distance	$b'_r - b'_f \geq 0$
		r-car doesn't stop	$v'_r > 0$
2	t_0,	non-safe distance	$d_{rss} > d_{real}$
		minimal distance	$d_{min} \leq d_{real}$
	$t_0+\varrho$,	r-car has a longer braking distance	$b'_r - b'_f \geq 0$
		r-car stops	$v'_r = 0$
3	t_0,	non-safe distance	$d_{rss} > d_{real}$
		minimal distance	$d_{min} \leq d_{real}$
		r-car has a longer braking distance	$b_r - b_f > 0$
		f-car very slow (may stop)	$v_f - a_{max,brake} * \varrho \leq 0$
	$t_0+\varrho$,	r-car has a shorter braking distance	$b'_r - b'_f < 0$
		f-car doesn't stop (braking or accelerating)	$v'_f > 0$
4	t_0,	non-safe distance	$d_{rss} > d_{real}$
		minimal distance	$d_{min} \leq d_{real}$
		r-car has a longer braking distance	$b_r - b_f > 0$
		f-car not very slow (can't stop)	$v_f - a_{max,brake} * \varrho > 0$
	$t_0+\varrho$,	r-car has a shorter braking distance	$b'_r - b'_f < 0$
		f-car doesn't stop (braking or accelerating)	$v'_f > 0$

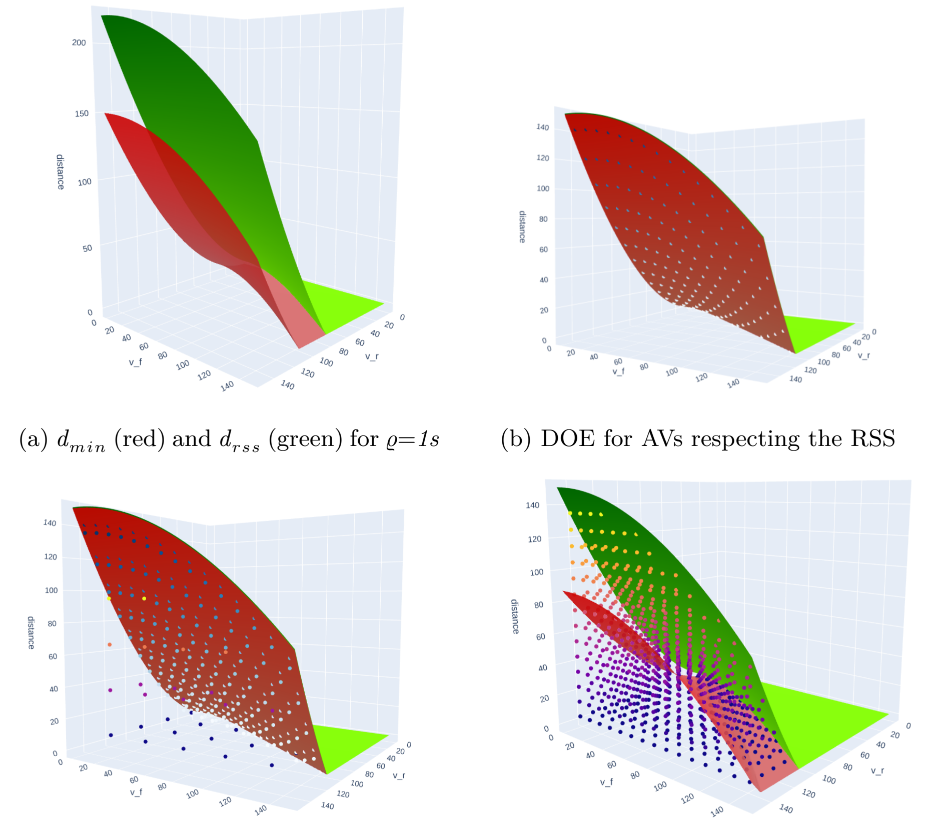
Table 1: Table with Abstract Test Cases for $drive_{rss}$

now represents a test selection grid which is located between the two surfaces. Density and starting points of the grid are configurable test parameters. Note that for realistic values of ϱ, such as *30 ms* for instance, the two surfaces are very close.

These surfaces can also be used for more realistic DOEs represented in figures 11c and 11d, relaxing the assumption that all other vehicles respect RSS. Figure 11c illustrates the need to test that the AV does apply $a_{min,brake}$ when distance to the front car lies right after the red surface ("tipping points") or drops suddenly (e.g. in case front car crashes). Figure 11d illustrates a DOE aiming at verifying if the AV applies some emergency braking between the green surface (usual RSS safety distance) and the red surface (RSS safety distance for the highest deceleration physically achievable).

5.3 Improvement of the Safe Distance Formula: RSS vs RSS$^+$

We want to illustrate the RSS and RSS$^+$ safe distance formulas, calculated on the basis of a set of plausible values for the constants and variables that compose them. We also want to estimate their relevance, by comparing them to some well-known safety distances: the *stopping distance* (d_{stop}, see [48], Section 2) and the *security distance* (d_{sec}, see [49]), corresponding to the "two-second rule" (see [50]).



(a) d_{min} (red) and d_{rss} (green) for ϱ=*1s*

(b) DOE for AVs respecting the RSS

(c) DOE for an AV respecting the RSS, among vehicles not respecting it

(d) DOE to analyse how an AV applies emergency braking

Fig. 11: Several ways to leverage the RSS safety distances to design DOEs

$$\begin{aligned} \rho &= 1\ \text{s and } 3 \times 10^{-2}\ \text{s} \\ a_{max,accel} &= 3.5\ \text{m}\,\text{s}^{-2} \\ a_{min,break} &= 5.8\ \text{m}\,\text{s}^{-2} \\ a_{max,break} &= 11\ \text{m}\,\text{s}^{-2} \end{aligned}$$

Reaction time values are for a typical human (ϱ=*1s*) and for an automated driving system (ϱ=*30ms*). Acceleration values are deduced from [48], Section 2.

ϱ	$v_r = v_f$	d_{rss}	d_{rss^+}	d_{rss^+}	d_{sec}	d_{stop}
(s)	(km/h)	(m)	(m)	(m)	(m)	(m)
			$a_r{=}0$	$a_r{=}-5.8$		
1	30	19	11	3	17	11
0.03	30	3	3	3	17	4
1	50	33	22	8	28	23
0.03	50	9	8	8	28	11
1	80	58	42	20	44	44
0.03	80	21	21	20	44	24
1	110	90	69	38	61	73
0.03	110	40	39	38	61	46
1	130	114	89	53	72	95
0.03	130	55	54	53	72	63

Table 2: Comparison of safety distance values computed with various formulas

Looking at Table 2, we can draw the following conclusions:

- The numerical application of the enhanced safety distance formula is rather low for the typical reaction time of an automated driving system.
- Interestingly, RSS$^+$ safe distance proves very relevant for human reaction times, which could be leveraged to raise warnings during periods of non-automated driving.

On the other hand, the RSS$^+$ formula produces an envelope which is smoother and thus tolerates less jolts from the driving strategies, as illustrated in Fig. **??**.

6 Conclusion

The growth of the autonomous vehicle industry is conditioned by its acceptance by the general public. Public support will largely depend on the confidence in the safety of these novel transport systems (cf. [9] for a survey). First standardization approaches targeting the development and implementation processes related to AVs attempt to address this need, most notably the ISO 21448 standard SOTIF.

In this paper, we present a number of novel modeling and $V\&V$ techniques intended to represent an alternative to widespread simulations, i. e. finite and usually very large sets of calculations of concrete, logical and abstract scenarios. Instead, we suggest semi-automated formal proof that can guarantee the exhaustive coverage of these scenario classes. By representing classes of scenarios by HOL-CSP processes (see Sect. 2.3), we can

- give a formal semantics for extensional scenarios and scenario classes, their various levels of abstraction and subsumption relationships (see Sect. 3);

- a rigorous (formal) definition of the safety goal in the validation activity itself on the conceptual and concrete level (see Sect. 4 and Sect. 5.1);
- a proof-based coverage criterion and a method to compose a covering set of abstract validation tests (see Sect. 5.2); and
- an improved driving strategy with regard to RSS (see Sect. 5.3).

Overall, we provide a groundwork to study the formalization, the analysis and the combination of driving strategies in AVs. We would like to stress that although we are critical of some aspects of the SOTIF in its present form, our results are compatible with its safety goals and concepts.

We believe that the exhaustive coverage is the adequate means to reduce "known hazards", and together with non-determinism and model extensionability our approach can reduce the "unknown hazards", too.˙

6.1 Related Work

Since the term has been coined around 2006 by the American Science Foundation, there have been numerous approaches to model and analyse Cyber-Physical Systems.

Simulation-based approaches, which are common practice in the emerging AV industry, suffer from a number of short-comings: First, simulators which are usually based on floating point calculations, have principally difficulties to cover non-continuous (e.g. bouncing) behaviour of physical systems. Depending on parameters and used abstractions, there is even the danger to overlook critical points in scenarios, and secondly, the question of exhaustiveness of simulations is often a major obstacle in formal certifications. Third, used abstractions tend to be ad-hoc and are difficult to justify, but without them complexity barriers are hit early. Forth, changes of models imply reruns of billions of scenarios which can slow down the development. In contrast, our approach covers the *complete* set of possible scenarios, computations are made symbolically on mathematical real numbers and reruns of our proofs due to changes take just a few minutes. By the way, these kind of limitations have been the reason for industrial partners to fund this study.

Model-checking based approaches require constructing a finite transitions system through a discrete abstraction such as a hybrid automaton. These approaches are based on computing a set of reachable states to automatically verify that the system satisfies a set of expected properties. Tools [3,24,19,20] are usually restricted to *linear* hybrid systems, or are limited to numerical approximation methods (see [21] for a survey). More recent approaches such as [10,18,27] use bounded model checking for reachability analysis to prove safety properties on these systems. Concerning the implementation side, these tools are direct implementations in some programming language, which is in contrast to the advantages of a conservative derivation approach. Moreover, all these approaches have seen only limited success due to lacking flexibility in modeling and too strong limitations with respect to proof power.

In order to overcome these limitations, and also profiting from the increasing maturity of implementing systems, proof-based approaches seem nowadays the most promising route towards the formal analysis of not-too-simple CPS. Offering more expressivity, models were commonly divided into *event-triggered* and *time-triggered* systems. The former let evolve a differential equation system up to a point where a particular condition, the *guard*, is met which gives back control to the discrete control system. The latter foresee an active sampling on the side of the discrete controller. While event-triggered models are reputed to be easier to specify, they are as such not implementable, which is the advantage of the latter. Obviously, our approach is in the time-triggered camp.

Platzer [5,35] suggests to use first-order dynamic logic (dL) used in an implementation of the KeYmaera system. Recent re-implementations are also based on a kernel architecture and a small axiomatization of a first-order fragment of linear differential equation theory (proven sound itself in a separate work in HOL [34,36]). Still, the trustbase of the KeYmaera system is significantly larger; HOL-CSP, in contrast, and the used `Isabelle/HOL-Analysis-` library are strictly definitional, therefore consistent to ZFC which is proven inside the system. Moreover, the fundamental restriction to linear differential equation theory excludes the possibility to reason over, for example, the floor-function $\lfloor x \rfloor$, which has to be modeled by continuous and differentiable approximations. In our view, $\lfloor x \rfloor$ is the key element in the modeling of the discretization-phase of CPS and its theory should be addressed directly rather than beeing abstracted away (an abstraction which will come at the price of a heavy foundational machinery). Finally, we find it important to contribute and profit from the open platform HOL and its fairly large user community makeing available thousands of system extensions (theories and components) in the Archive of Formal Proofs [15]. It is this community effort which makes the large scale verification efforts possible, that range from physics to the nitty-gritty details of controller hardware, albeit still based on safe logical extension principles. Last but not least, [35] and subsequent publications suggest to model communications between actors via game theory; we prefer classical analysis of deadlock and lifeness properties of concurrent systems along the lines of [43]. Either way, these two modeling approaches will boil down to the construction of a global invariant of the concurrent system which will be notoriously difficult. Both frameworks provide a refinement notion based either on symbolic trajectories or symbolic traces; the precise relationship between these two notions needs further investigation.

Differential Hoare Logics (dH) [17] is another proof-based approach offering also differential refinement using Isabelle/HOL. Properties like ordinary differential equation liveness or program correctness were stated in dH, broken down into (simpler) step-by-step refinements using dR and proved in Isabelle. The approach uses the Kleene algebras [1] with tests and the Morgan-style approach to derive rules for verification condition generation and refinement laws of dR. The authors have developed specific support [32] for the modeling and verification tasks.

And finally, Event-B and the Rodin platform have been proposed to formalize CPS as abstract machines which were refined to (still rather abstract) implementations [41,8,14]. The approach suffers from the relatively weak automated proof support and the necessary axiomatizations of the mathematic foundations.

Car control is a deep area that has been studied by various different communities; consequently, there is a growing body of literature on collision avoidance algorithms for AVs. Some approaches propose discrete-time point based temporal logics or discrete time automata [30,4]. Substantial abstractions on the dynamics of AVs have to be made in these approaches. Other approaches are based on trajectory planning an checking, based on reachable sets [26,31] in ODE‘s; while these approaches take quite complex dynamics into account, it seems unclear if this approach leads to sufficiently robust and fast algorithms that can be used "on-board" in an AV. An early major case study on "Adaptive Cruise Control" [29] treats the problem of distributed car control system; unfortunately, the accompanying technical report contains only a fairly abstract presentation of the proof work done.

As far as testing is concerned, by contrast with [23], which uses RSS to select relevant scenarios among variations of a specific scenario for a given automated system, we aim at defining *a priori* the whole subspace of relevant (hazardous) scenarios for a wide family of safety controllers (namely, all such systems with perfect sensors and relying on detection of actors restricted to their current position and speed).

6.2 Future Work

We are currently generalizing RSS and its variants to two-dimensional topologies and study the combinations of RSS and variants thereof with topology-aware driving strategies (curved lanes, crossings, etc.). Other possible directions are the inclusion of models for noise on the global scene, in order to go beyond the unrealistic "captors are perfect" assumption. And furthermore, the kinematics model might be refined admitting external forces and resistances, as suggested in [28].

An interesting future extension is the use of the Isabelle code generation facilities, which have been used to randomly generate traces from CSP-like languages[16]. This way, HOL-CSP could be used to *generate* simulators directly from scenario processes. This could help to improve the acceptance of our approach by traditional engineers in the AV domain, and speed up the process of validation by different means than proof.

And finally: a fascinating extension of this work is the refinement of models capturing driving strategies to actual controller code in a realistic programming language such as C. Again, the versatility of the Isabelle/HOL platform offers the potential to make this work: components with semantic theories for C offering sufficient proof automation such as [22] are available. Thus, a seamless transition from high-level CPS models down to realistic implementations can be assured, based solely on theories constructed in a definitional, logically safe way.

Acknowledgment

This work has been supported by the French government under the ”France 2030” program, as part of the SystemX Technological Research Institute.

References

1. Armstrong, A., Gomes, V.B.F., Struth, G.: Building program construction and verification tools from algebraic principles. Formal Aspects Comput. **28**(2), 265–293 (2016). https://doi.org/10.1007/s00165-015-0343-1, https://doi.org/10.1007/s00165-015-0343-1
2. ASAM: ASAM OpenSCENARIO V2.0.0-PRC.1, §6.3.1 Levels of scenario abstraction (Dec 2021), https://asam-ev.github.io/public_release_candidate/asam-openscenario/2.0.0/conceptual-overview/scenario-abstraction.html
3. Asarin, E., Dang, T., Maler, O.: The d/dt tool for verification of hybrid systems. In: Brinksma, E., Larsen, K.G. (eds.) Computer Aided Verification. pp. 365–370. Springer Berlin Heidelberg, Berlin, Heidelberg (2002)
4. Bannour, B., Niol, J., Crisafulli, P.: Symbolic model-based design and generation of logical scenarios for autonomous vehicles validation. In: IEEE Intelligent Vehicles Symposium, IV 2021, Nagoya, Japan, July 11-17, 2021. pp. 215–222. IEEE (2021). https://doi.org/10.1109/IV48863.2021.9575528, https://doi.org/10.1109/IV48863.2021.9575528
5. Beckert, B., Platzer, A.: Dynamic logic with non-rigid functions. In: Furbach, U., Shankar, N. (eds.) Automated Reasoning (IJCAR). LNCS, vol. 4130, pp. 266–280. Springer (2006). https://doi.org/10.1007/11814771_23
6. Brookes, S.D., Hoare, C.A.R., Roscoe, A.W.: A theory of communicating sequential processes. J. ACM **31**(3), 560–599 (1984)
7. Brookes, S.D., Roscoe, A.W.: An improved failures model for communicating sequential processes. In: Brookes, S.D., Roscoe, A.W., Winskel, G. (eds.) Seminar on Concurrency. pp. 281–305. Springer, Berlin, Heidelberg (1985)
8. Butler, M.J., Abrial, J., Banach, R.: Modelling and refining hybrid systems in event-b and rodin. In: Petre, L., Sekerinski, E. (eds.) From Action Systems to Distributed Systems - The Refinement Approach, pp. 29–42. Chapman and Hall/CRC (2016). https://doi.org/10.1201/b20053-5, https://doi.org/10.1201/b20053-5
9. Cartenì, A.: The acceptability value of autonomous vehicles: A quantitative analysis of the willingness to pay for shared autonomous vehicles (SAVs) mobility services. Transportation Research Interdisciplinary Perspectives **8**, 100224 (10 2020). https://doi.org/10.1016/j.trip.2020.100224
10. Chen, X., Ábrahám, E., Sankaranarayanan, S.: Flow*: An analyzer for non-linear hybrid systems. In: Sharygina, N., Veith, H. (eds.) Computer Aided Verification. pp. 258–263. Springer Berlin Heidelberg, Berlin, Heidelberg (2013)
11. Crisafulli, P., Taha, S., Wolff, B.: Modelling and Proving Safety in Autonomous Cars Scenarios in HOL-CSP. Research report, University Paris-Saclay ; IRT SystemX, Palaiseau (Oct 2021), https://hal.inria.fr/hal-03429597
12. Derler, P., Lee, E.A., Sangiovanni-Vincentelli, A.: Modeling Cyber-Physical Systems. Proceedings of the IEEE (special issue on CPS) **100**(1), 13 – 28 (January 2012), http://chess.eecs.berkeley.edu/pubs/843.html

13. Dick, J., Faivre, A.: Automating the generation and sequencing of test cases from model-based specications. In: Woodcock, J., Larsen, P. (eds.) Formal Methods Europe 93: Industrial-Strength Formal Methods. LNCS, vol. 670, pp. 268–284. Springer (Apr 1993)
14. Dupont, G., Ameur, Y.A., Singh, N.K., Pantel, M.: Formally verified architectural patterns of hybrid systems using proof and refinement with event-b. Sci. Comput. Program. **216**, 102765 (2022). `https://doi.org/10.1016/j.scico.2021.102765`, `https://doi.org/10.1016/j.scico.2021.102765`
15. Eberl, M., Klein, G., Lochbihler, A., Paulson, L., Nipkow, T., Thiemann(eds.), R.: Archive of Formal Proofs (afp) (2022), `https://www.isa-afp.org/`
16. Foster, S., Hur, C., Woodcock, J.: Formally verified simulations of state-rich processes using interaction trees in isabelle/hol. In: Haddad, S., Varacca, D. (eds.) 32nd International Conference on Concurrency Theory, CONCUR 2021, August 24-27, 2021, Virtual Conference. LIPIcs, vol. 203, pp. 20:1–20:18. Schloss Dagstuhl - Leibniz-Zentrum für Informatik (2021). `https://doi.org/10.4230/LIPIcs.CONCUR.2021.20`, `https://doi.org/10.4230/LIPIcs.CONCUR.2021.20`
17. Foster, S., y Munive, J.J.H., Struth, G.: Differential Hoare Logics and Refinement Calculi for Hybrid Systems with Isabelle/HOL. In: Fahrenberg, U., Jipsen, P., Winter, M. (eds.) Relational and Algebraic Methods in Computer Science - 18th International Conference, RAMiCS 2020, Palaiseau, France, April 8-11, 2020, Proceedings [postponed]. Lecture Notes in Computer Science, vol. 12062, pp. 169–186. Springer (2020). `https://doi.org/10.1007/978-3-030-43520-2\_11`, `https://doi.org/10.1007/978-3-030-43520-2_11`
18. Fränzle, M., Herde, C., Teige, T., Ratschan, S., Schubert, T.: Efficient Solving of Large Non-linear Arithmetic Constraint Systems with Complex Boolean Structure. J. Satisf. Boolean Model. Comput. **1**(3-4), 209–236 (2007). `https://doi.org/10.3233/sat190012`, `https://doi.org/10.3233/sat190012`
19. Frehse, G.: PHAVer: algorithmic verification of hybrid systems past HyTech **10**(3), 263–279 (2008). `https://doi.org/10.1007/s10009-007-0062-x`
20. Frehse, G., Le Guernic, C., Donzé, A., Cotton, S., Ray, R., Lebeltel, O., Ripado, R., Girard, A., Dang, T., Maler, O.: SpaceEx: Scalable Verification of Hybrid Systems. In: Ganesh Gopalakrishnan, S.Q. (ed.) Proc. 23rd International Conference on Computer Aided Verification (CAV). LNCS, Springer (2011)
21. Geretti, L., Sandretto, J.A.D., Althoff, M., Benet, L., Collins, P., Duggirala, P., Forets, M., Kim, E., Mitsch, S., Schilling, C., Wetzlinger, M.: Arch-comp22 category report: Continuous and hybrid systems with nonlinear dynamics. In: Frehse, G., Althoff, M., Schoitsch, E., Guiochet, J. (eds.) Proceedings of 9th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH22). EPiC Series in Computing, vol. 90, pp. 86–112. EasyChair (2022). `https://doi.org/10.29007/fnzc`, `https://easychair.org/publications/paper/JrQ4`
22. Greenaway, D., Lim, J., Andronick, J., Klein, G.: Don't sweat the small stuff: Formal verification of C code without the pain. In: ACM SIGPLAN Conference on Programming Language Design and Implementation. pp. 429–439. ACM, Edinburgh, UK (Jun 2014). `https://doi.org/10.1145/2594291.2594296`
23. Hekmatnejad, M., Hoxha, B., Fainekos, G.: Search-based Test-Case Generation by Monitoring Responsibility Safety Rules (2020). `https://doi.org/10.48550/ARXIV.2005.00326`, `https://arxiv.org/abs/2005.00326`
24. Henzinger, T.A., Ho, P.H., Wong-Toi, H.: Hytech: A model checker for hybrid systems. In: Grumberg, O. (ed.) Computer Aided Verification. pp. 460–463. Springer Berlin Heidelberg, Berlin, Heidelberg (1997)

25. Hoare, C.A.R.: Communicating Sequential Processes. Prentice-Hall, Inc., Upper Saddle River, NJ, USA (1985)
26. Kochdumper, N., Gassert, P., Althoff, M.: Verification of collision avoidance for commonroad traffic scenarios. In: Frehse, G., Althoff, M. (eds.) 8th International Workshop on Applied Verification of Continuous and Hybrid Systems (ARCH21), Brussels, Belgium, July 9, 2021. EPiC Series in Computing, vol. 80, pp. 184–194. EasyChair (2021). https://doi.org/10.29007/1973, https://doi.org/10.29007/1973
27. Kong, S., Gao, S., Chen, W., Clarke, E.: dreach: δ-reachability analysis for hybrid systems. In: Baier, C., Tinelli, C. (eds.) Tools and Algorithms for the Construction and Analysis of Systems. pp. 200–205. Springer Berlin Heidelberg, Berlin, Heidelberg (2015)
28. Koopman, P., Osyk, B., Weast, J.: Autonomous Vehicles Meet the Physical World: RSS, Variability, Uncertainty, and Proving Safety (Expanded Version) (2019). https://doi.org/10.48550/ARXIV.1911.01207, https://arxiv.org/abs/1911.01207
29. Loos, S.M., Platzer, A., Nistor, L.: Adaptive cruise control: Hybrid, distributed, and now formally verified. In: Butler, M.J., Schulte, W. (eds.) FM 2011: Formal Methods - 17th International Symposium on Formal Methods, Limerick, Ireland, June 20-24, 2011. Proceedings. Lecture Notes in Computer Science, vol. 6664, pp. 42–56. Springer (2011). https://doi.org/10.1007/978-3-642-21437-0_6, https://doi.org/10.1007/978-3-642-21437-0_6
30. Maierhofer, S., Moosbrugger, P., Althoff, M.: Formalization of intersection traffic rules in temporal logic. In: 2022 IEEE Intelligent Vehicles Symposium, IV 2022, Aachen, Germany, June 4-9, 2022. pp. 1135–1144. IEEE (2022). https://doi.org/10.1109/IV51971.2022.9827153, https://doi.org/10.1109/IV51971.2022.9827153
31. Manzinger, S., Pek, C., Althoff, M.: Using reachable sets for trajectory planning of automated vehicles. IEEE Trans. Intell. Veh. **6**(2), 232–248 (2021). https://doi.org/10.1109/TIV.2020.3017342, https://doi.org/10.1109/TIV.2020.3017342
32. Huerta y Munive, J., Struth, G.: Predicate transformer semantics for hybrid systems: Verification components for isabelle/hol. Journal of Automated Reasoning **66** (02 2022). https://doi.org/10.1007/s10817-021-09607-x
33. Nipkow, T., Paulson, L.C., Wenzel, M.: Isabelle/HOL—A Proof Assistant for Higher-Order Logic, LNCS, vol. 2283. Springer (2002). https://doi.org/10.1007/3-540-45949-9
34. Platzer, A.: A Complete Axiomatization of Quantified Differential Dynamic Logic for Distributed Hybrid Systems. Logical Methods in Computer Science **Volume 8, Issue 4** (Nov 2012). https://doi.org/10.2168/LMCS-8(4:17)2012, https://lmcs.episciences.org/720
35. Platzer, A.: Logical Foundations of Cyber-Physical Systems. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-63588-0
36. Platzer, A., Tan, Y.K.: Differential equation axiomatization: The impressive power of differential ghosts. In: Dawar, A., Grädel, E. (eds.) Proceedings of the 33rd Annual ACM/IEEE Symposium on Logic in Computer Science, LICS 2018, Oxford, UK, July 09-12, 2018. pp. 819–828. ACM (2018). https://doi.org/10.1145/3209108.3209147, https://doi.org/10.1145/3209108.3209147
37. Rashid, A., Siddique, U., Hasan, O.: Formal verification of platoon control strategies. In: Johnsen, E.B., Schaefer, I. (eds.) Software Engineering and Formal Methods - 16th International Conference, SEFM 2018, Held as Part of STAF 2018,

Toulouse, France, June 27-29, 2018, Proceedings. Lecture Notes in Computer Science, vol. 10886, pp. 223–238. Springer (2018). https://doi.org/10.1007/978-3-319-92970-5_14, https://doi.org/10.1007/978-3-319-92970-5_14

38. Roscoe, A.: Theory and Practice of Concurrency. Prentice Hall (1997)
39. Scott, D.: Continuous lattices. In: Lawvere, F.W. (ed.) Toposes, Algebraic Geometry and Logic. pp. 97–136. Springer, Berlin, Heidelberg (1972)
40. Shalev-Shwartz, S., Shammah, S., Shashua, A.: On a Formal Model of Safe and Scalable Self-driving Cars. arXiv e-prints arXiv:1708.06374 (Aug 2017)
41. Su, W., Abrial, J., Zhu, H.: Formalizing hybrid systems with event-b and the rodin platform. Sci. Comput. Program. **94**, 164–202 (2014). https://doi.org/10.1016/j.scico.2014.04.015, https://doi.org/10.1016/j.scico.2014.04.015
42. Taha, S., Ye, L., Wolff, B.: HOL-CSP Version 2.0. Archive of Formal Proofs (Apr 2019), http://isa-afp.org/entries/HOL-CSP.html
43. Taha, S., Ye, L., Wolff, B.: Philosophers may Dine - Definitively! In: Furia, C.A. (ed.) Integrated Formal Methods (iFM). No. 12546 in Lecture Notes in Computer Science, Springer-Verlag, Heidelberg (2020). https://doi.org/10.1007/978-3-030-63461-2_23
44. Technical Committee ISO/TC 22, Subcommittee SC 32: Road vehicles — Safety of the intended functionality. techreport ISO 21448:2021, International Organization for Standardization, https://www.iso.org/standard/77490.html
45. Tej, H., Wolff, B.: A corrected failure divergence model for CSP in Isabelle/HOL. In: Fitzgerald, J.S., Jones, C.B., Lucas, P. (eds.) Formal Methods Europe (FME). LNCS, vol. 1313, pp. 318–337. Springer (1997). https://doi.org/10.1007/3-540-63533-5_17
46. Ulbrich, S., Menzel, T., Reschka, A., Schuldt, F., Maurer, M.: Defining and Substantiating the Terms Scene, Situation, and Scenario for Automated Driving. In: 2015 IEEE 18th International Conference on Intelligent Transportation Systems. pp. 982–988. IEEE, IEEE, Las Palmas, Spain, http://ieeexplore.ieee.org/document/7313256/https://www.researchgate.net/profile/Andreas_Reschka/publication/283726201_Defining_and_Substantiating_the_Terms_Scene_Situation_and_Scenario_for_Automated_Driving/links/5653044608ae4988a7af37b0/Defining-and-Su
47. (UNSW), T.S.: Autocorres: automatic specification abstraction (2022), https://trustworthy.systems/projects/TS/autocorres/
48. Wikipedia: Distance d'arrêt, https://fr.wikipedia.org/w/index.php?title=Distance_d%27arr%C3%AAt&oldid=181727336
49. Wikipedia: Distance de sécurité en france, https://fr.wikipedia.org/w/index.php?title=Distance_de_s%C3%A9curit%C3%A9_en_France&oldid=193199417
50. Wikipedia: Two-second rule, https://en.wikipedia.org/w/index.php?title=Two-second_rule&oldid=1089170388