Integrating Formal Verification and Simulation-based Assertion Checking in a Corroborative V&V Process

Maike Schwammberger[1], Christopher Harper[2], Gleifer Vaz Alves[3], Greg Chance[4], Tony Pipe[2], Kerstin Eder[4]

[1] University of Oldenburg, Oldenburg, Germany
[2] University of the West of England, Bristol, UK
[3] Federal University of Technology, Parana, Brazil
[4] University of Bristol, Bristol, UK

***Abstract*–Automated Vehicles (AVs) are rapidly maturing in the transportation domain. However, the complexity of the AV design problem is such that no single technique is sufficient to provide adequate validation of key properties such as safety, reliability or trustworthiness. In this vision paper, a combination of a spatial traffic logic and agent-based verification methods with a validation method that uses assertion checking of simulations is proposed. We sketch how to integrate the respective approaches within a methodological framework called Corroborative Verification and Validation (V&V). The Corroborative V&V framework identifies three different verification and validation levels for AVs (formal verification, simulation-based testing, real-world experiments) and specifies connections and evidence between these levels. We define specifications for the formal relationships that must be established between processes, system models and requirements models for the evidence from formal design verification and simulation-based testing to corroborate each other and enhance assurance confidence from verification and validation.**

1 Introduction

As a future with Automated Vehicles (AVs) comes closer each day, it is of the utmost importance to ensure their safety, reliability and trustworthiness. In this paper, we present our vision to investigate methods for combining formal verification with simulation-based testing to establish an integrated methodology for the safety validation of AVs. We envision to apply a verification and validation philosophy called Corroborative Verification and Validation (V&V), an approach developed previously at the Bristol Robotics Laboratory [17]. The Corroborative V&V seeks to provide verification and validation assurance through the transformation and cross-checking of verification and validation evidence at different abstraction levels of design description and modelling.

Complex highly automated systems such as AVs have many safety properties that must be validated prior to their entry into service. While design verification by formal proof of properties provides high-quality assurance evidence, proofs are generally only performed for individual properties and require significant effort to perform. Formal analyses of multiple properties together can become very complex, requiring major effort. For this reason, we plan to investigate how to use individual property proofs to identify edge cases for simulation-based testing, to show that the individual properties hold as a set or to investigate system behaviour at boundary conditions or where properties may conflict with one another.

We plan to investigate the integration of formal design verification methods applied to AVs, using techniques such as Urban Multi-lane Spatial Logic (UMLSL) [15], developed at the University of Oldenburg, and agent-based verification methods [2], developed at Federal University of Technology – Parana, with methods for developing assertion checks developed at the Bristol Robotics Laboratory [11] for extracting simulation test cases from textual requirements such as "the rules of the road" documents (for example, the UK Highway Code (UKHC) [8]). For this, we also make use of a road traffic rule extension of UMLSL called Urban Spatial Logic for Traffic Rules (USL-TR), with which some UK Traffic Rules have been formalised [16].

Formal verification and simulation can be thought of as complementary tools or techniques for checking system correctness with respect to a specification. Some authors have used simulators to build or determine variable values for [12] formal models of complex systems such as AVs. During simulation-based verification the parameter space is 'sampled' but for formal verification the entire space is 'proven' against a single property. Goldberg [10] attempts to integrate these techniques through use of the satisfia-

Statements about authorship contribution. Maike Schwammberger (e-mail: schwammberger@informatik.uni-oldenburg.de) is with the University of Oldenburg, Oldenburg, Germany. M. Schwammberger was supported by the German Research Council (DFG) in the PIRE Projects SD-SSCPS and ISCE-ACPS under grant no. FR 2715/4-1 and FR 2715/5-1. Christopher Harper (e-mail: chris.harper@brl.ac.uk), Tony Pipe (e-mail: tony.pipe@brl.ac.uk), are with the University of the West of England, Frenchay, Coldharbour Ln, Bristol, BS34 8QZ, United Kingdom. Gleifer Vaz Alves (e-mail: gleifer@utfpr.edu.br), is with the Federal University of Technology, Parana, Brazil. Greg Chance (e-mail: greg.chance@bristol.ac.uk), and Kerstin Eder (e-mail: kerstin.eder@bristol.ac.uk) are with the Trustworthy Systems Lab, Department of Computer Science, University of Bristol, Merchant Ventures Building, Woodland Road, Bristol, BS8 1UQ, United Kingdom. G. Chance and K. Eder were supported by the "UKRI Trustworthy Autonomous Systems Node in Functionality" under grant number EP/V026518/1.

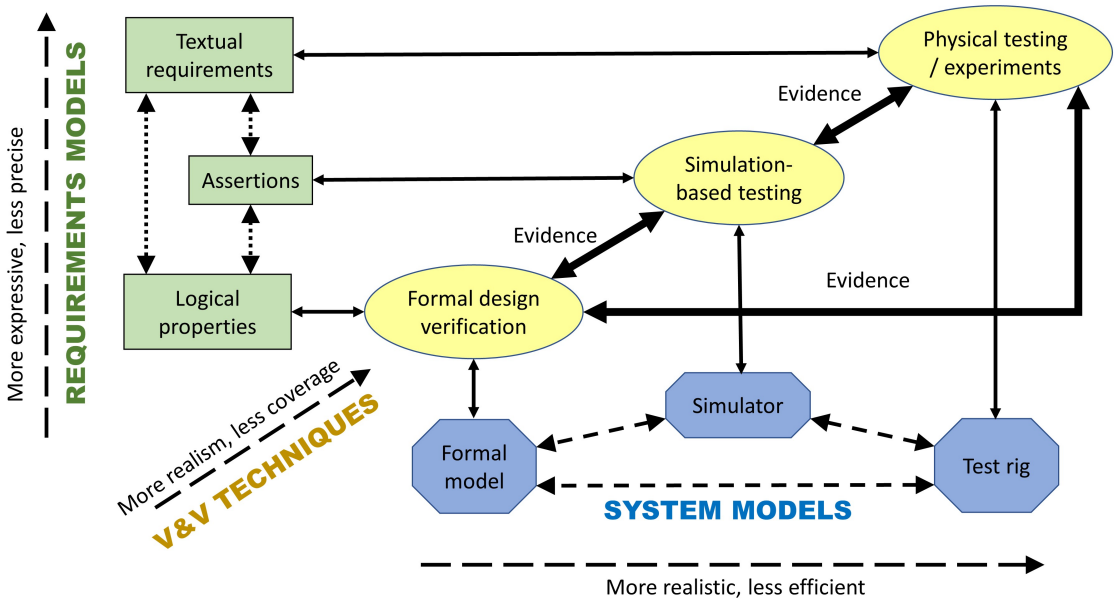

Fig. 1: Lifecycle process model for Corroborative V&V, adapted based on [17].

bility problem to derive a sufficient sample set that has 'enough power' to prove the logical satisfiability of the simulation model with respect to a formal model. There is also empirical evidence of the corroboration of simulation and formal techniques to support the verification for a self-driving application [9] and in a robot handover task [17]. Yet Chen et al. [7] present the integration of tools developed for railway systems, the *BRaVE* tool, a railway simulator, and *OnTrack*, a tool used for automatic verification of safety properties for scheme plans using formal methods. The authors intend to use a single environment for prototyping, concept, development and safety analysis.

Our contribution is structured as follows. In Sect. 2, we give an overview of the Corroborative V&V philosophy and in Sect. 3, we briefly introduce the spatial traffic rule logic USL-TR and some steps into the direction of the verification of timed-automata traffic rule controllers. We provide details on assertion checking of simulations in Sect. 4, followed by a sketch of our vision to combine the approaches from Sects. 3 and 4 in Sect. 5 under the roof of the Corroborative V&V methodology from Sect. 2.

2 Corroborative V&V

The problem of automated vehicle V&V is complicated by the fact that validation of situated behaviour in extended domains requires a very large state space, in which all the relative states between the AV and features of the environment (other agents, static features, ambient conditions, etc.) must be defined. This leads to combinatorial expansion in the number of states to be covered, which rapidly becomes impracticable to test to acceptable coverage levels (even with automated testing). Therefore, it is critical to select specific test cases that deliver important information about the system properties of interest such as safety or reliability rather than simply performing tests mechanically across the state space.

One intention of Corroborative V&V, as illustrated in Fig. 1, is to assist the process of identifying significant test cases by using information derived from V&V activities at different levels of design refinement to cross-check each other, and provide a means of revealing the test cases that are more significant to the validation of key system properties. The framework presented in Fig. 1 considers three levels of V&V activity: static formal verification of designs by model checking or theorem proving, simulation-based testing, and experiments with physical test rigs or road test vehicles.

The principal trade-off between these levels of activity lies between coverage and realism. At the level of systems design, formal verification of system properties can achieve exhaustive coverage over the domains over which a design is proven. However, typically only individual properties are proven, and proofs are often grounded in assumptions or other axiomatic statements that can only be validated externally to the proof itself. At the level of the complete system, physical tests have the advantage of being realistic and therefore provide direct evidence of system safety properties. But the scope of this evidence is limited only to the system and environmental states covered in the specific test cases performed, and typically the cost and time involved in the achievement of effective coverage of all conditions are either too expensive or time-consuming to be practicable.

Simulation-based testing lies between these two levels of capability; in providing a dynamic test of a complete system model it can evaluate multiple system properties simultaneously (using assertion checking techniques [11]) over any range of environmental conditions that can be simulated, and automated testing over large sets of test

`https://en.wikipedia.org/wiki/Boolean_satisfiability_problem`

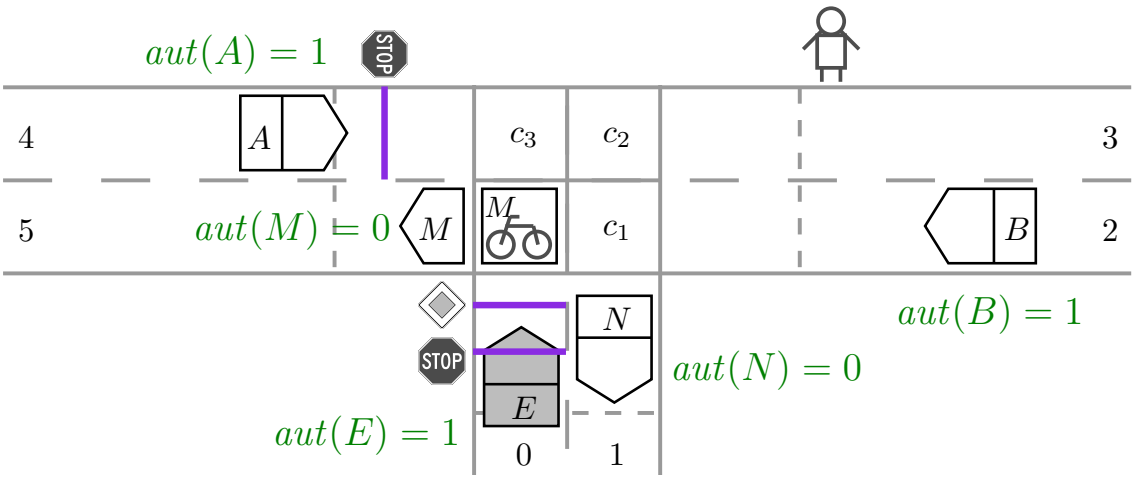

Fig. 2: An example traffic situation showing traffic signs and several automated ($aut(E) = 1$) and non-automated ($aut(E) = 0$) road users A, B, E, M and N.

conditions can be performed. Such evaluation does not constitute formal proof, but specific simulation tests could reveal conditions that challenge or overturn proofs generated at the design verification stage, especially in any underlying axiomatic statements (assumptions). The principal difficulty with simulation lies in the fidelity of the simulator [6]; inaccuracies in modelling the real world may lead to mis-evaluation of system properties, possibly leading to false-positive claims that they are valid. So, the behaviour of systems in simulation must be verified against physical test data to validate the simulation.

By using each type of V&V process judiciously, to provide cross-correction of the others, the disadvantages of any one can be compensated by the advantages of the others, and the overall quality of evidence can result in enhanced assurance of safety or any other important system property. In this paper we discuss the issues involved in integrating formal design verification with simulation-based testing; integration with physical testing is left for future work.

3 From a Traffic Logic to the Verification of Autonomous Agents

We intend to build automated agents capable of knowing and recognising road traffic rules. For that, we define several steps, starting with the representation of traffic rules using a specific logic, USL-TR [16], designed to abstract the existent elements of traffic rules as they are written in a given Highway Code [8]. With USL-TR, it is possible to express spatial elements of traffic rules. Such spatial elements could, e.g., be that there is a stop sign or a pedestrian crossing the road ahead.

We consider the traffic situation depicted in Fig. 2 and the AV, termed the ego car E, as a running example. Now take a simple fragment of the traffic rule (**170**) from the UKHC [8]: "*Do not cross or join a road **until** there is a **gap** large enough for you to do so safely.*" Using USL-TR, we abstract the spatial elements of this formula. For instance, a safe gap for the ego car E can be represented by $sg(E) \equiv free \wedge \ell >= size_E$, where *free* indicates that there is free space on the road, and $\ell >= size_E$ specifies that this free space is larger than the size of E. Using a spatial connector $\frown$ from USL-TR, similar to the chop operator ; of Interval Temporal Logic [13], we could then express that this free space is, e.g., in front of the "reservation" (the occupied space) of E with a formula $re(E) \frown sg(E)$. In Fig. 2, this formula would not hold, as there is not enough free space in front of E due to the road user M currently occupying the intersection. With an "autonomy flag" $aut(E) = 1$ it is indicated that E is an automated road user and $aut(M) = 0$ states that M is a non-automated road user (e.g. a cyclist).

In current work [3], we consider the temporal sequence of elements in a traffic rule by representing them using behaviour diagrams. At this stage, we do not yet add formal details to these diagrams but reflect the flow of actions of the rules as they are. These diagrams can be perceived as an intermediate step towards a fully formalised traffic rule. By annotating the behaviour diagrams with USL-TR formulae, we reach the first step for our planned translation from behaviour diagrams into timed automata [1]. The following step is to add clock constraints for expressing timing behaviour contained within traffic rules (e.g. that braking or setting a turn signal are not done immediately but take some time).

Applying the previous steps, we obtain a formal, timed-automaton representation of traffic rules. From this, we build the corresponding autonomous agents endowed with traffic rules. We will then apply model checking techniques to these agents and formally verify properties, both for the timed automata and autonomous agents, starting from previous work in [4] and [2]. Retaking the previous example, we could check properties such as "*Always when there is a stop sign, the agent first stops and proceeds with their manoeuvre when there is a safe gap large enough to do so*". Our overall goal is to introduce a *Digital Highway Code* that comprises a digitalised, precise and unambiguous version of the existing natural language road traffic regulations for AVs.

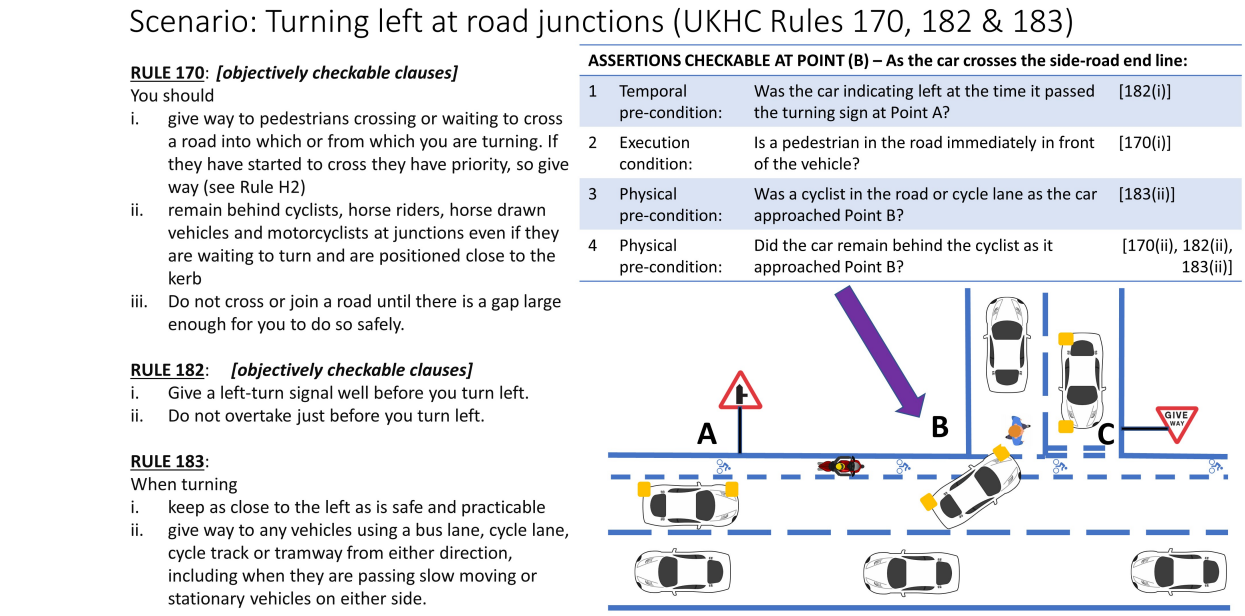

Fig. 3: Example assertions based on UK Highway Code rules.

4 Validation by Assertion Checking of Simulations

In previous work [11] we investigated the methodology of deriving safety validation assertions from driving manuals or codes of practice, specifically the UKHC. We developed two distinct methods for constructing assertions: direct translation of natural language, and model-based analysis.

Many UKHC assertions define independently observable conditions or situations of the ego car being tested, and can be measured objectively. Assertions of this type are usually defined as conditions relative to some *assertion reference point*, a time-step in the simulation at which the trigger for the rule can or should be applied. We have found that these assertions can be classified into one of four principal types:

- *Invariant Condition*: A condition to be satisfied at all time steps within the captured data trace.
- *Execution Condition*: A condition to be satisfied at the assertion reference point.
- *Pre-condition (physical or temporal)*: A (physical or temporal) condition to be satisfied in the steps preceding the assertion reference point.
- *Post-condition (physical or temporal)*: A (physical or temporal) condition to be satisfied in the steps following the assertion reference point.

The following example in Fig. 3 shows the identification of several assertion hypotheses derived from UKHC rule 170 and others relating to the same road environment presented in Section 3 and in Fig. 2:

As discussed in [11], the hypotheses are then transformed into queries and can be executed in a database in which the simulation data trace has been stored. The illustration shows how several different assertions may apply in any given scenario. While an automated vehicle control system design may have been verified to satisfy each rule in isolation, assertion checking in simulation can verify whether all the verified properties are satisfied together in the scenario. If an assertion is triggered at any time step, the causes of the assertion failure can be examined in the data.

5 Vision - Corroborating Design Verification with Simulation Testing

Our motivation for this paper is to investigate how to integrate the approaches that have been sketched in the previous Sects. 3 and 4 using the Corroborative V&V framework from Sect. 2. The goal of the framework is to provide evidence from each of the V&V processes, such that evidence generated by one process supports evidence generated by the others. When taken together, confidence in the overall safety (or other properties) is improved. Taking the subset of elements of Fig. 1 related to these two processes, and expanding the inter-relationships (arrows), we consider the following lattice diagram to capture the precise relationships between models, requirements and processes:

Our objective is to define what is needed to ensure that evidence from Formal Design Verification corroborates that of Simulation-based Testing, and vice versa. In general parlance, the term “corroborate” means ‘to provide information in support of’ some statement; this can be taken to mean that if evidence from one process is found to be consistent with another, then the two corroborate one another. The corroborative V&V framework was used in [17] in such a manner. However, the term has been applied by Popper [14] to mean that *an attempt has been made to falsify the evidence, which has failed*, thereby offering stronger support for the claims supported. This is a stronger criterion, and we propose to investigate how this

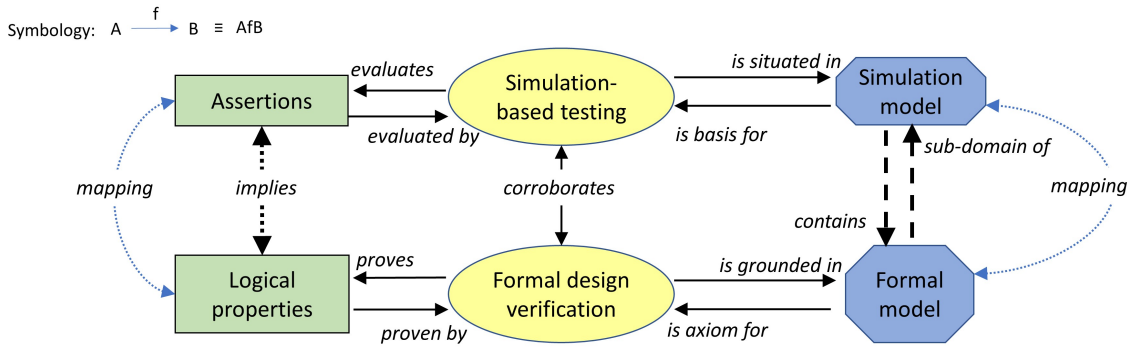

Fig. 4: Relationships between Formal Design Verification and Simulation-based Testing

may be achieved within the Corroborative V&V Framework. Possibilities include:

- Testing at boundary conditions of any assumptions defined as axioms in the formal design verification (e.g. road network, traffic conditions constraints).
- Testing where different (possibly conflicting) actions are defined for similar model conditions by proofs of different properties (so-called "trolley problems" [5]).
- Testing at boundary conditions of initial conditions within which safety properties have been formally verified.
- For the planned agent model-checking (cf. Sect. 3), it is of high interest to find a meaningful, maximally representative model to start with, which may be found via simulations.
- Other cases where different formal properties may conflict.

By combining our simulation- and verification-based approaches within the Corroborative V&V framework via the sketched steps, our vision is that both types of approaches will be able to support ("corroborate") each other via evidence that is exchanged between them.

References

[1] R. Alur and D. L. Dill. "A Theory of Timed Automata". In: *Theoretical Computer Science* 126.2 (1994), pp. 183–235.

[2] G. V. Alves, L. Dennis, and M. Fisher. "A Double-Level Model Checking Approach for an Agent-Based Autonomous Vehicle and Road Junction Regulations". In: *Journal of Sensor and Actuator Networks* 10.3 (2021). URL: https://www.mdpi.com/2224-2708/10/3/41.

[3] G. V. Alves and M. Schwammberger. *Towards a Digital Highway Code using Formal Modelling and Verification of Timed Automata*. submitted to FMAS'22.

[4] C. Bischopink and M. Schwammberger. "Verification of Fair Controllers for Urban Traffic Manoeuvres at Intersections". In: *Formal Methods. FM 2019 International Workshops - Porto, Portugal, October 7-11, 2019, Revised Selected Papers, Part I*. Vol. 12232. Lecture Notes in Computer Science. Springer, 2019, pp. 249–264.

[5] J.-F. Bonnefon, A. Shariff, and I. Rahwan. "The social dilemma of autonomous vehicles". In: *Science* 352.6293 (2016), pp. 1573–1576. eprint: http://science.sciencemag.org/content/352/6293/1573.full.pdf. URL: http://science.sciencemag.org/content/352/6293/1573.

[6] G. Chance, A. Ghobrial, K. McAreavey, S. Lemaignan, T. Pipe, and K. Eder. "On Determinism of Game Engines Used for Simulation-Based Autonomous Vehicle Verification". In: *IEEE Transactions on Intelligent Transportation Systems* (2022), pp. 1–15.

[7] L. Chen, P. James, D. Kirkwood, H. N. Nguyen, G. L. Nicholson, and M. Roggenbach. "Towards integrated simulation and formal verification of rail yard designs - an experience report based on the UK East Coast Main Line". In: *2016 IEEE International Conference on Intelligent Rail Transportation (ICIRT)*. 2016, pp. 347–355.

[8] Department for Transport. *Using the road (159 to 203) - The Highway Code - Guidance - GOV.UK*. en. 2017. URL: https://www.gov.uk/guidance/the-highway-code/using-the-road-159-to-203 (visited on 04/13/2018).

[9] A. Domenici, A. Fagiolini, and M. Palmieri. "Integrated simulation and formal verification of a simple autonomous vehicle". In: *International Conference on Software Engineering and Formal Methods*. Springer. 2017, pp. 300–314.

[10] E. Goldberg. "On bridging simulation and formal verification". In: *International Workshop on Verification, Model Checking, and Abstract Interpretation*. Springer. 2008, pp. 127–141.

[11] C. Harper, G. Chance, A. Ghobrial, S. Alam, T. Pipe, and K. Eder. "Safety Validation of Autonomous Vehicles using Assertion-based Oracles". In: (2021). arXiv: 2111.04611. URL: https://arxiv.org/abs/2111.04611.

[12] R. Hoffmann, M. Ireland, A. Miller, G. Norman, and S. Veres. "Autonomous agent behaviour modelled in PRISM–A case study". In: *International Symposium on Model Checking Software*. Springer. 2016, pp. 104–110.

[13] B. Moszkowski. “A Temporal Logic for Multilevel Reasoning About Hardware”. In: *Computer* 18.2 (1985), pp. 10–19.

[14] K. Popper. *The logic of scientific discovery*. Routledge, 1997.

[15] M. Schwammberger. “An abstract model for proving safety of autonomous urban traffic”. In: *Theoretical Computer Science* 744 (2018), pp. 143–169.

[16] M. Schwammberger and G. V. Alves. “Extending Urban Multi-Lane Spatial Logic to Formalise Road Junction Rules”. In: *Electronic Proceedings in Theoretical Computer Science* 348 (Oct. 2021). arXiv: 2110.12583, pp. 1–19. URL: `http://arxiv.org/abs/2110.12583` (visited on 11/07/2021).

[17] M. Webster et al. “A corroborative approach to verification and validation of human-robot teams”. In: *Int. J. Robotics Res.* 39.1 (2020).