Testing, Validation, and Verification of Robotic and Autonomous Systems: A Systematic Review

HUGO ARAUJO, Universidade Federal de Pernambuco
MOHAMMAD REZA MOUSAVI, University of Leicester
MAHSA VARSHOSAZ, IT University of Copenhagen

We perform a systematic literature review on testing, validation, and verification of robotic and autonomous systems. The scope of our review covers all testing interventions that address the system-level qualities of robotic and autonomous systems; they include peer-reviewed research papers proposing new techniques, improving or evaluating the applicability of existing techniques, by considering their information sources (particularly, models and data-sets), people, processes, and tools.

Our survey is performed based on a rigorous methodology structured in three phases: in the first phase, starting from a set of 26 seed papers (selected by domain experts) and the SERP-TEST taxonomy, through coding the seed papers we both refined our taxonomy to cover the specific aspects of testing, validation. and verification for robotic and autonomous systems and designed the first version of our search query and . In the second phase, we retrieved 3651 papers from three different academic search engines and after the first round of review and applying inclusion / exclusion criteria 512 were further considered and after reviewing the full papers, 109 papers were selected and coded. In the third phase, we used the results of the second phase to validate, refine, and extend the search query. The third search resulted in 7478 unique papers; we finally included and coded 202 papers using our refined taxonomy for testing RAS.

Our objective is to answer four research questions, pertaining to (1) the type of models, (2) measures for efficiency, effectiveness, and adequacy, (3) tools and their availability, and (4) evidence of applicability, particularly in industrial contexts. Our intended audience are researchers and practitioners working in the domain of testing robotic and autonomous systems. To give a more refined analysis of the results, we classify the results in a number of sub-categories including the application domain, cooperation and connectivity, and the testing strategy. We analyse the results of our coding to answer our research questions and identify strengths and gaps in this domain and present two sets of recommendations for researchers and practitioners.

CCS Concepts: • **Software and its engineering → Formal software verification**.

ACM Reference Format:
Hugo Araujo, Mohammad Reza Mousavi, and Mahsa Varshosaz. 2021. Testing, Validation, and Verification of Robotic and Autonomous Systems: A Systematic Review . 1, 1 (January 2021), 39 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

Authors' addresses: Hugo Araujo Universidade Federal de Pernambuco, hlsa@cin.ufpe.br; Mohammad Reza Mousavi University of Leicester, mm789@leicester.ac.uk; Mahsa VarshosazIT University of Copenhagen, mahv@itu.dk.

1 INTRODUCTION

1.1 Motivation

Robotic and Autonomous Systems (RAS) involve a rich integration of several disciplines such as control engineering and robotics, mechanical engineering, electronics, and software engineering. Validation and verification of RAS entails a non-trivial extension of traditional testing techniques to deal with their multi-disciplinary nature. In particular, for researchers and practitioners from the software testing community, extending the existing software testing techniques to RAS is a challenge that has led to a sizeable literature on proposing and evaluating different techniques and processes. This rich literature calls for a secondary study that puts a structure to this landscape and identifies relative strengths and weakness of available results. The present paper addresses this gap by performing a structured literature survey of RAS testing.

There are a number of earlier surveys on related topics; we provide an in-depth comparison of related work with our survey in Section 2. However, briefly speaking, some of these surveys have a different or more confined scope, e.g., considering machine learning components [68], formal specification and verification techniques [116] or driving data-sets [88], or do not aim to provide a structure overview of the field in order to answer concrete questions for a given audience [20]. To our knowledge, this is the first secondary study that covers the breadth of interventions for testing RAS (see the Related Work section for other studies with different foci) and moreover, provides an analysis of such results with the aim of characterising the type of intervention and analyse their evidence of applicability (in terms of tools and large-scale case study).

1.2 Scope and audience

Our scope covers novel interventions and research results (including techniques, process, tools, and applications thereof) that deal with testing robotic and autonomous systems. We provide a precise definition of RAS in the remainder of this paper, in order to derive rigorous inclusion and exclusion criteria. However, our scope naturally excludes interventions that focus on a specific unit or component of such systems (e.g., a specific type of learning or planning algorithms or testing of physical or mechanical parts of such systems) and do not take the system-level validation and verification into account.

Our audience are both researchers in software and systems engineering who are looking for gaps and strengths in this area in order to inform their future research as well as practitioners who are looking for interventions that are applicable to their specific problems. Next, we define a number of research questions that help us structure and analyse the existing interventions for these two groups of audience.

For our interventions to be useful for the intended audience, we confine our scope to those interventions that

(1) address testing the computer systems integrated in RAS (as opposed to only physical, mechanical, or control parts) in their methodology; this is justified by the fact that our intended audience are researchers and practitioners in software and systems engineering, and
(2) have some evidence of applicability, efficiency, or effectiveness on RAS; this is motivated by our scope (validation and verification of RAS) as well as our goal to provide evidence of strength (or weakness) for researchers and practitioners.

1.3 Research questions

As specified above, we would like to review and analyse those interventions that are applicable to validation and verification of RAS; in particular, we have an emphasis on those intervention that take into account the computer systems in RAS and their interactions with their physical environment and human users. We would like to perform our analysis from two perspectives:

(1) researchers: to identify strengths and gaps in the research landscape of testing RAS, particularly concerning the traditional software testing taxonomies, are there new challenges not covered by software testing taxonomies, and
(2) practitioners: identify interventions that have the evidence of applicability for a given environment and given resources.

Our research questions are specified below:

(1) What are the *types of models* used for testing RAS?
(2) Which *efficiency, effectiveness* and *coverage measures* were introduced or used to evaluate RAS testing interventions?
(3) What are the interventions supported by *(publicly available) tools* in this domain?
(4) Which interventions have *evidence of applicability* to large-scale and industrial systems?

1.4 Structure of the Paper

The remainder of this paper is structured as follows. In Section 2, we review related work, with a focus on secondary studies (literature surveys and reviews) on related subject matters. In Section 3, we define the scope of the paper and explain the background to this structured review. There, we report on the core set of results we started with as the seed for our search in order to shape the study. In Section 4, we review the strategy we used for the our systematic review; this include the description of our search and selection strategy, the development of the taxonomoy used for coding the results, our data extraction and synthesis methods. In this section, we also reflect on the threats to our study. In Section 5, we present the results of our coding and analyse them to answer our research questions. In Section 6, we reflect on our analysis and provide concrete suggestions for our target audience, i.e., both for researchers and practitioners. In Section 7, we conclude the paper and present some directions of future research.

2 RELATED WORK

There are a number of literature reviews, surveys, and mapping studies conducted which cover different aspects of robotic and autonomous systems. In what follows, we give an overview of the ones that are most related and have the closest connection to our study (in chronological order).

Cortesi, Ferrara, Chaki [48] discuss the features of a number of static analysis techniques, namely data-flow analysis, control-flow analysis, model-checking and abstract interpretation. The survey covers features such as automation, precision, scalability, and soundness for these techniques. The goal for the study is stated as providing robotics software developers hints to help choosing appropriate analysis approaches depending on the kind of properties of interest and software system. However, the interventions studied in this paper are not necessarily applied in the robotics domain already. Furthermore, the work is not a systematic review and does not claim providing any coverage on existing work on analysis techniques applied in its target application domain.

Helle, Schamai and Strobel [81] as well as Redfield and Seto [153] provide an overview of challenges in and available techniques and results for testing and verification autonomous systems. Both studies only sample a small subset of available results and techniques and use them to identify the areas requiring future research. Our findings based on a much larger set, provides a much more refined view about the available interventions and the landscape for future research.

Koopman and Wagner [97] give an overview of challenges in the V model adapted to deal with the problems in the context of autonomous vehicles. The paper identifies five major challenge areas in testing according to the V model for autonomous vehicles, namely, driver out of the loop, complex requirements, non-deterministic algorithms, inductive learning algorithms, and fail operational systems. The paper covers solution approaches that seem promising across these different challenges including phased deployment using successively relaxed operational scenarios, and using a monitor/actuator pair architecture to separate complex autonomy functions from simpler safety functions, and fault injection. Similar to the previous two papers, the work of Koopman and Wagner has a more restrictive scope than the present paper; moreover, the above-mentioned work is not a (structured) review of the literature.

Gao and Tan [68] provide an overview of the state-of-the-art in V&V for safety-critical systems that rely on machine learning techniques (based on deep learning) for autonomous driving. In this work, the researchers first extract a set of studies by conducting a search and identify a set of challenges by reviewing these studies. Then, the validity of the identified challenges is checked by setting up an industrial questionnaire to survey. Furthermore, a set of research recommendations are provided for future work in automated driving based on deep learning. The search query used in this study is more limited than ours in scope, because it focuses on testing for automated driving and deep learning, while we cover robotic and autonomous systems in a much broader sense. The articles covered in this study are published before 2017.

Knauss et al. [94] present an empirical study for investigating software-related challenges of testing automated vehicles. In the work two different kinds of data collection namely, focus groups (including eleven practitioners from Sweden) and interviews (including 15 practitioners and researchers from a number of countries) are used. The work provides insights about challenges such as virtual testing and simulation, standards and certifications, increased need to test nonfunctional aspects, and automation. This work is not a systematic mapping.

Rao and Frtunikj [152] identify three concrete issues regarding assessment of functional safety of neural networks used in automotive industry intending to initiate the discussion with industrial peers to find practical solutions. The issues include: dataset completeness, neural network implementation, and transfer learning.

Kang, Yin, and Berger [88] provide a survey of publicly available driving datasets as well as virtual testing for autonomous driving algorithms. A detailed overview of 37 datasets for open-loop testing and 22 virtual testing environments for closed-loop testing have been provided. A remarkable aspect of this survey is the involvement of an industrial domain expert. The scope and results of the paper is significantly different from ours: they focus on autonomous driving algorithms, while we include the whole domain of RAS; they focus on datasets and tools, while we focus on interventions and their effects, as well as their tools.

Beglerovic, Metzner, and Horn [20] provide a brief overview of methodologies used for testing in automated driving. The work provides recommendations about promising methodologies and research areas aimed to reduce the testing effort. The authors mention challenges such as complexity of automated driving functions, variation of scenarios and parameters, scenario selection and test generation. Furthermore, the work briefly touches upon validation, supporting tools in the validation task, and standardisation. This paper is significantly different in methodology from ours: it is not a mapping study and does not provide any detail about the coverage of existing work.

Luckcuck et al. [116] provide a survey of formal specifications and verification methods and tools used for autonomous robotics systems. The work covers a range of studies from 2007-2018. In this work, a number of challenges for formally modelling and verifying the environments that the robotic systems operate in in addition to the internals of such systems is provided. The work differs from ours as it only covers formal specification and verification tools for such systems. Hence, techniques such as (non-exhaustive) testing and simulation are not covered in this work. Also, our work has a different methodological approach in that we pose and answer research questions as the result of our secondary study, while they focus on the literature review itself. We did use the studies reviewed by Luckcuck et al. to validate and refine our search query in the third phase of our research.

Rajabali et al. [150] perform an extensive and systematic literature review on software validation and verification for autonomous vehicles. Their scope is more restricted than the scope of the present study, but some of their research questions (such as identifying gaps in the literature) are common to ours. However, their methodology does not involve a detailed taxonomy as in the present study and hence, their conclusions are more abstract and at a higher level. We have also used this recent paper to validate the query and the final set of considered papers in the third phase of our research.

3 BACKGROUND AND RATIONALE

In this section, we provide an overview of the motivation behind this literature survey, and define its domain. Subsequently, we introduce the basic taxonomy that we have extended and adapted for coding the literature. We also review the pilot study that was used to shape our taxonomy (and later validate our search query, presented in the next section).

3.1 Motivation

Based on our study of the existing literature reviews and surveys, we identified the gap for a secondary study that 1) presents a structured review of the existing results on validation and verification of robotic and autonomous systems and 2) targets specific research questions regarding a) the types of models, b) measures of efficiency and effectiveness, c) available tools, and d) evidence of applicability to large-scale and industrial systems.

3.2 Robotic and Autonomous System

There are a variety of definitions for our domain, RAS; these definitions encompass aspects such as autonomy (including high-level decision making and planning), and adaptation (including artificial intelligence and machine learning) and interaction with human users and the physical environment (including perception, actuation, and mobility). In our view, the following definition provides a concise synthesis of these aspects:

> An autonomous system is an intelligent system that is designed to deal with the physical environment on its own, and work for extended periods of time without explicit human intervention. They are built to analyse, learn from and act on the surrounding environment.

We emphasise two important aspects of this definition: one is the system-level perspective; hence, modules or units of software and hardware that are not autonomous systems themselves will not be included in our studies; the second important aspect is the interaction with the environment; hence, autonomous systems that work on offline data and do not feature an interaction with their environment are excluded as well.

3.3 Testing and the SERP-Test Taxonomy

We consider testing as any structured approach to *validate* or *verify* the quality of a robotic and autonomous system. Validation concerns checking the system specification, design, or implementation against user requirements. Verification concerns checking the system specification, design, or implementation against another piece of specification, design, or implementation. In other words, validation checks whether we have built the right system (for its users), while verification checks whether we have built it correctly (with respect to other specifications and artefacts) [145].

Our classification of testing research is based on the SERP-Test taxonomy [58]. This taxonomy provides a very general framework for classifying and communicating software testing research and have been used and adapted for this purpose across different domains [3, 155]. In SERP-Test, testing research is classified in terms of four facets: intervention, effect, scope, and context. Intervention pertains to the test techniques, their adaptation, and adoption in different context. Effect facet is used to identify the improvement or adaptation in a given practise as well as any insights gained through assessment. The scope specified whether the effect has been materialised in planning, design, execution, or analysis of tests. Context, as its name suggests, specifies the environment where the intervention takes place, in terms of people and their knowledge, the system under test, and the required models and other types of information.

In the next section, we report on the methodology of this study; namely, in Section 4.1 we discuss the seed papers that formed the basis of our search, in Section 4.2, we report on the search query and its validation with respect to the seed papers; in Section 4.3, we report on the final inclusion and exclusion criteria; and in Section 4.4, we report on the adapted taxonomy. Finally, in Section 4.5 we detail our strategy to extract data from the set of included papers.

4 METHODOLOGY

In this section, we present the methodology used throughout our study that encompasses three phases. In the first phase, a pilot study was conducted, in which, we gathered a set of seed papers, extracted keywords to form our search query, developed a set of inclusion/exclusion criteria and refined our taxonomy. In the second phase, we performed the search, applied the exclusion criteria and coded the selected papers. In the final phase, the search query was validated and refined via an analysis of the secondary studies on the subject; a new search was performed and additional studies were included for review and coding.

A repository containing artefacts of this study (namely, the seed papers, the result of the searches and the coding) is publicly available[1].

4.1 Seed papers

The set of seed papers contain 26 manually selected studies gathered in consultation with domain experts from both academia and industrial partners. We reviewed this set as a pilot study with the following objectives:

(1) gathering keywords for the initial search query,
(2) sharpening the inclusion and exclusion criteria, and
(3) evaluating and adapting the SERP-Test taxonomy.

4.2 Search strategy

From the seed papers, an initial set of keywords was extracted to form a search query; additional terms with close meanings and relation to the initial keywords were used to broaden the search. In addition to Google Scholar, two digital

[1] https://bit.ly/2XavlUR

libraries, namely, ACM and IEEE, that broadly cover publications with topics in computer science and engineering fields, have been selected as search venue.

Our query is a conjunction of two main sub-queries: one that comprises terms relevant to our application domain, robotic and autonomous systems, and the other, contains the terms related to testing and verification. The initial query was as follows.

("Robots" OR "Robotics" OR "Deep learning" OR "Machine Learning" OR "Artificial Intelligence" OR "Robot Simulator" OR "Autonomous Vehicle" OR "Autonomous Vehicles" OR "Autonomous Cars" OR "Image Classification Systems" OR "Neural Networks" OR "Unmanned Vehicles" OR "Unmanned Aerial Vehicles" OR "UAV" OR "Connected and Autonomous Vehicle" OR "CAV" OR "Automated Functions" OR "Drive Assist" OR "Multi-Agent Systems" OR "Autonomous Agents")
AND
("Testing" OR "Validation" OR "Verification" OR "Safety Case Analysis" OR "Runtime Monitoring" OR "Robustness" OR "Simulation" OR "Coverage" OR "Metaheuristics" OR "Search-Based" OR "Combinatorial" OR "SMT Solving" OR "SAT Solving" OR "Constraint Solving" OR "Model Checking")

For this first search, we limited the scope of our search to papers published between 2008 and 2019. Its outcome was a set of 3030 studies. A review of their titles led to the identification and exclusion of the majority of the results. Only 523 studies were classified as potentially relevant for this survey.

In the last and final phase of our research, we made use of the seed papers and secondary studies to validate the outcome of the above-given search. Then, we refined the query by including missing keywords and removing keywords that did not result in included papers. Furthermore, we observed that from years 2008 to 2014, only a handful of papers were included; this led us to further focus the search to papers published between 2014 and 2019. The new search resulted in a total of 7478 unique papers (i.e., the duplicates from the first search were automatically excluded), out of which, 194 were classified as relevant after title review. The final search query is as follows:

("Robots" OR "Robot" OR "Robotics" OR "Robotic" OR "Swarm" OR "Swarms" OR "Autonomous" OR "Unmanned" OR "UAV" OR "UAVs" OR "CAV" OR "Automated Functions" OR "Automated Driving" OR "Drive Assist" OR "Multi-Agent Systems" OR "Multi-Agent System" OR "Driverless" OR "Self-Driving" OR "ADAS")
AND
("Testing" OR "Validation" OR "Verification" OR "Verifying" OR "Verifiably" OR "Safety Case Analysis" OR "Runtime Monitoring" OR "Metaheuristics" OR "Simulation" OR "SMT Solving" OR "SAT Solving" OR "Constraint Solving" OR "Model Checking" OR "Search-Based")

All in all, the current research is the result of considering a total of 10,534 papers, with only 717 being selected for further full paper review and a total of 202 were finally included in our study based on the criteria that is discussed next.

4.3 Selection strategy

To set the boundaries for the scope of our study, based on our research questions, we defined and used a set of inclusion/exclusion criteria as follows.

4.3.1 Inclusion Criteria. The criteria considered for inclusion of studies is as follows:

- The topic of the study is on Testing RAS (Robotics and Autonomous Systems)
- The context must consider the cyber and physical aspects of a system (as opposed to only physical, mechanical, or control parts.)
- Evidence for applicability is provided

In the scope of our study, *Testing* is interpreted in a broad sense, which includes formal verification techniques, static and dynamic testing, validation and non-exhaustive techniques.

4.3.2 Exclusion Criteria. The studies matching the following criteria are excluded.

- Not available online.
- Not in English.
- Short papers
- Not peer-reviewed
- Patents
- Published before 2008 (in the second phase), published before 2014 (in the third phase)
- Not addressing robotics and autonomous systems
- No research contributions to testing (incl. validation or verification)
- Only testing units in isolation; not considering the robotics and autonomous systems as a whole. (If the contribution for testing units are not specific to the system considered in the paper and can have applications in the bigger context, we included the study.)
- The study only considers the physical aspects of the system and not software components,
- Concerning human-controlled systems, e.g., UAVs and robots that are remotely controlled by a human.
- For papers on the topic of simulation, as there are a large number of studies among the search results which do not have new contributions in the process or technique of testing interventions; we consider excluding such papers unless they provide clear contributions in the context of testing, validation and verification, have available tool, or provide evidence of applicability in industrial context.

We applied our inclusion and exclusion criteria to the set of 717 papers that have been selected in the previous step. As a result, a total of 202 studies were filtered for the next step, i.e., review and coding.

4.4 Taxonomy

In order to consistently classify the set of included studies to extract the information required for answering the research questions described in Section 1.3, we follow a modified version of the SERP-Test taxonomy (see Section 3.3. We started with the high-level facets proposed in SERP-TEST taxonomy and throughout a number of iterations we defined and re-defined a number of categories based on the information obtained from coding the included studies. The final taxonomy, based on which the studies are classified, is as follows.

Context. For context we consider two main categories, namely, system under test and the technique.

- **System Under Test.** System under test describes the type of systems on which the testing technique is applied. In our study, we consider two main categories of RAS, namely, *Robotics* and *Autonomous Systems*. These two categories of systems are selected as they dominate the case studies and a broad range of systems that are considered in the studies concerning testing RAS.
- **Technique.** This is the second category that is considered under Context which represents the testing technique that is improved or affected as a part of the contributions of the work to testing RAS.
- **Models** Different types of models can be used for describing the behaviour of a system under test. We consider this category to extract the information about the variety of models that are used in the work on testing RAS.
- **Tools and languages** This category consists of details on tools and languages, under which, the subject systems are described.

Effect. We refine the *Effect* facet (see Section 3.3) further to four categories as follows.

- **Metrics.** This category encompasses the metrics used as a way of evaluating test adequacy or correctness of the subject, based on performance (i.e., efficiency and effectiveness) or coverage measures.
  - **Performance.** This category describes the effect of an intervention on the performance of the testing technique or the subject system. The performance covers a variety of measures concerning the time and resources required to perform testing.
  - **Coverage.** This category concern the measures that indicate how comprehensive the testing technique is once performing in the context of RAS.
- **Process.** This category describes the kind of effects that impact the process of testing technique.
- **Technique.** The technique concerns methods presented as new testing methods or improvements for testing RAS.
- **Tooling** In this category we extract the information about the type of tools that have been used throughout each work. We further classify the tools according to their availability: (1) open source, which are tools for which the source artefacts are available, (2) publicly available, which are tools that are accessible to be used but the source code has not been provided and (3) private, which are tools that have not been made available for download or purchase.

Scope. This facet in SERP-Test taxonomy is further refined to two main categories as follows:

- **Model testing.** This category represents techniques which use a model of the system for testing. We define two sub-categories for such techniques:
  - **Simulation.** This category comprises different types of simulation techniques used for testing RAS.
  - **Formal verification.** This category describes formal verification techniques that use a model of the system to rigorously verify the behaviour.
- **System testing.** This category describes techniques which are applied on actual implementation artifacts of systems.
  - **Static testing.** This category describes techniques which perform testing of system without code execution.
  - **Dynamic testing.** This category describes techniques which check the functional behaviour by executing the implemented code for the system.

Evaluation. We define three main subcategories for this facet in SERP-Test taxonomy (see Sect. 3.3)

- **Case Study.** This category specifies the type of systems that has been used in evaluations of the selected papers. We categorise the case studies into three subcategories, namely, small scale, benchmark, and industrial.
  - **Small.** We consider examples that are developed solely for the purpose of evaluating the method in a specific study and are not applicable for evaluating other similar intervention (due to lack of available details, lack of genericity, or insufficient scale / number of subject systems) as small scale.
  - **Benchmark.** We consider a case study as benchmark if it represents a set of systems with sufficient level of details such that they are / can be used as a point of reference in the evaluations performed in the context of testing autonomous systems.
  - **Industrial.** We categorise a case study as industrial if the subject system is of industrial scale and the evaluation has been performed in industrial context.

4.5 Data extraction strategy

As discussed in Section 4.2, a total of 717 papers have been selected as potentially relevant after title review. We further filtered out additional papers based on their abstracts.

Our data extraction methodology consisted of going through selected studies and applying the exclusion criteria given in Section 4.3. Each paper has been analysed by at least two of the authors. This led to a total of 202 studies that were coded according to our taxonomy and reviewed in detail as a part of this survey.

Figure 1 shows a summary of the number of published articles clustered by year of release. We notice a steady yearly increase of studies included in our review.

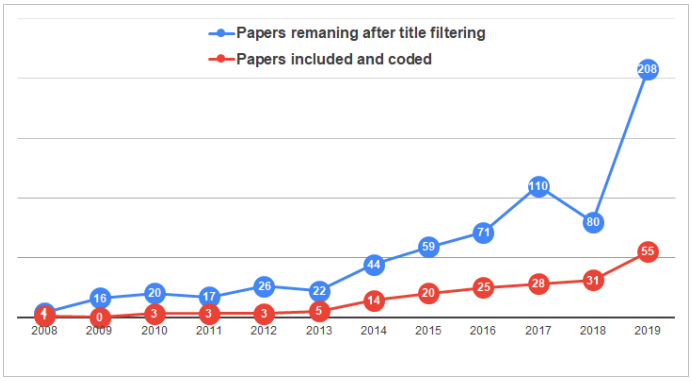

Fig. 1. Relevant and included papers by year.

5 RESULTS

In this section, we present the results of coding the literature in our taxonomy. We structure our results in terms of the four research questions. Regarding RQ1, we present the results concerning the different property specification languages and modelling languages and frameworks used for testing TAS. Regarding RQ2, we review the metrics used to measure the effectiveness, efficiency, and adequacy of testing interventions as well as the quality of systems under test. Regarding RQ3, we code the tools used to implement different interventions as well as any tools implementing the interventions themselves. Regarding RQ4, we present the evidence provided for applicability of the interventions in terms of the case studies and benchmarks used to evaluate the interventions.

5.1 RQ1: Models

In this section, we review the type of models and formalisms that are used for describing the behaviour of robotics and autonomous systems and their properties in testing interventions. Tables 1 and 2, show an overview of results of coding for models used in the studies included in this survey. We classified models into quantitative, qualitative, formal and informal. It should be clarified that, we consider a model to be quantitative, if it can represent measurable quantities such as real valued entities. Otherwise the model is considered qualitative. This is regardless of whether the results of the evaluation or the testing technique applied on the model is qualitative or quantitative.

5.1.1 Modelling Properties. As presented in Table 1, among all studies included in our survey, less than one third use a model or logic to describe the properties of the subject systems. For this set of studies all models are classified as formal.

Among those, we notice that, over two thirds employ logics to describe qualitative properties of systems [8, 9, 17, 19, 22, 23, 23, 36, 36, 55, 55, 59, 61, 65, 66, 70, 84, 87, 90, 98–101, 112, 114, 118, 132, 142, 147, 168, 174, 185, 189, 191, 192]. Linear temporal logic, temporal epistemic logic, epistemic alternating temporal logic are examples of such logics that have been used in this set of studies. The rest of studies use logics that can describe quantitative properties, e.g., describing stochastic or temporal aspects of systems [7, 8, 16, 33, 54, 74, 78, 90, 113, 115, 138, 143, 172, 203, 204].

Table 1. Model for system properties. The table includes studies which use different type of models, the classification of model as qualitative, quantitative, formal and informal and the domain in which the model can be used.

Papers	Model	Model type	Formalism	Domain
[7, 33, 143, 203, 204]	Probabilistic Computational Tree Logic (PCTL)	Stochastic	Formal	Agnostic
[8]	First-order logic	Qualitative	Formal	Agnostic
[54, 74]	Signal Temporal Logic (STL)	Time	Formal	Agnostic
[138]	Metric Interval Temporal Logic (MITL)	Time	Formal	Agnostic
[9, 17, 19, 36, 55, 65, 66, 84, 87, 118, 132, 142, 147, 191, 192]	Linear Temporal Logic (LTL)	Qualitative	Formal	Agnostic
[113]	Probabilistic Linear Temporal Logic (PLTL)	Stochastic	Formal	Agnostic
[70]	Graphical Structuring Notation (GSN)	Qualitative	Informal	Agnostic
[59, 174]	Computation Tree Logic (CTL)	Qualitative	Formal	Agnostic
[16, 78, 172]	Timed Computation Tree Logic (TCTL)	Time	Formal	Agnostic
[115]	Continuous stochastic logic	Stochastic	Formal	Agnostic
[36]	ForSALE	Qualitative	Formal	Agnostic
[61]	Property Specification Language (PSL)	Qualitative	Formal	Agnostic
[90]	First Order LTL	Time	Formal	Agnostic
[23, 98–101, 112, 114]	Epistemic temporal logics (ATL, ATLK, ACTL*K\X , IACTLK\X , CTLK)	Qualitative	Formal	Agnostic
[23]	Epstemic strategy logic (ESL)	Qualitative	Formal	Agnostic
[22]	Parameterised Data-Aware Multi-Agent Systems (P-DAMAS)	Qualitative	Formal	Agnostic
[168]	Temporal Logic of Actions	Qualitative	Formal	Agnostic
[185]	TRIO (Temporal Logic)	Qualitative	Formal	Agnostic
[55]	Rewriting logic	Qualitative	Formal	Agnostic
[189]	Past time linear temporal logic (ptLTL)	Qualitative	Formal	Agnostic

A review of the results presented in Table 1 shows there is a limited number of studies which consider analysis of properties of systems formulated using formal logics. Furthermore, quantitative properties are considerably less represented in the selected studies. Properties to verify stochastic, continuous and temporal aspects of the systems should play an important role when testing complex and real-time systems, such as RAS. This gap may emphasise the need for quantitative logics that are tailored for the domain.

5.1.2 Modelling System Behaviour or Structure. In Table 2, an overview of models used for describing the behaviour or structure of robotics and autonomous systems is provided. About half of the selected studies employ some sort of behavioural or structural model in their testing strategy. Mathematically defined models, i.e., formal models, are used in most of such interventions . For instance, Petri Nets and a variety of their extensions [10, 19, 65, 163, 199], labelled transition systems and some of their extended versions [12, 33, 78, 115, 161], finite state machines and their extensions [77, 118, 118], and Markov chains [18, 131, 143, 170, 203, 204] are examples of such models. One observation is that among studies which use informal description of systems, models that are used in Gazebo and on ROS are more commonly used [13, 14, 37, 37, 44, 83, 96, 105].

In some of the studies, a combination of models are used throughout the testing intervention; in particular, for some higher-level models, lower-level models are used to specify their semantics [40, 130].

Around half of those studies that consider a behavioural model of the system, use qualitative models. The rest of the studies use models which describe different quantitative aspects of systems such as time and stochastic behaviour, e.g., using timed Petri nets and stochastic Petri nets [19, 65], probabilistic timed automata [12, 115], and Markov chains [18, 131, 143, 170, 203, 204]; system dynamics, using differential equations [6, 47, 60, 107, 117, 124, 138, 173], hybrid automata and their extensions [34, 35, 71, 194–196], functional mockup units [1] and various informal simulation models for dynamical systems [13, 14, 37, 44, 83, 96, 105, 144, 171, 201].

Compared to studies before 2019, we notice that there has been a rise in using stochastic models. However, this number is small given that there are a variety of models and testing techniques introduced for describing the stochastic and probabilistic behaviour of systems in the testing community and that the stochastic and probabilistic behaviour are important in RAS; Hence, this indicates a need for models for describing probabilistic and stochastic behaviour of robotics and autonomous systems, which are tailored for the domain. Furthermore, we observe that qualitative models are still prevalent despite the importance of quantitative aspects of behaviour in RAS.

Table 2. Models for system behaviour or structure. The table includes studies which use different type of models to specify the behavioural or structure of the system, system models are further classified in terms as qualitative vs. quantitative, formal vs. informal and domain-specific vs. domain agnostic.

Papers	Model	Model type	Formalism	Domain
[33]	Labelled Transition System (LTS)	Qualitative	Formal	Agnostic
[199]	Predicate Transition Re-configurable Nets (PrTR Nets)	Qualitative	Formal	Agnostic
[13, 14, 52, 53, 62, 87, 137]	Belief Desire Intention (BDI)	Qualitative	Formal	Agnostic
[13, 14, 37, 44, 83, 96, 105]	Unified Robot Description Format (URDF)	Continuous Dynamics	Informal	Robotics
[6, 47, 60, 107, 117, 124, 138, 173]	Differential Equations	Continuous Dynamics	Formal	Agnostic
[144]	USARsim (Unreal Script)	Continuous Dynamics	Informal	Robotics
[51]	Meta-model (UML-like)	Qualitative	Informal	Agnostic
[7, 16, 63, 78, 161]	Timed Automata	Time	Formal	Agnostic
[9, 55, 80, 118, 147, 159]	Finite State Machine (FSM)	Qualitative	Formal	Agnostic
[70]	Graphical Structuring Notation (GSN)	Qualitative	Informal	Agnostic

Continued on next page

Table 2 – *Continued from previous page*

Papers	Model	Model type	Formalism	Domain
[29, 111, 168, 174]	Extended Finite State Machine	Time, Priority	Formal	Agnostic
[25, 27, 135]	Matlab/Simulink models	Time	Informal	Agnostic
[123]	Prolog	Qualitative	Formal	Agnostic
[18, 131, 143, 170, 203, 204]	Markov chain	Stochastic	Formal	Agnostic
[15]	Continuous time Markov chain	Time, Stochastic	Formal	Agnostic
[132]	Markov Decision Processes (MDP)	Stochastic	Formal	Agnostic
[182]	Series-Parallel Action Graphs	Qualitative	Formal	Agnostic
[125]	Alloy	Qualitative	Formal	Agnostic
[187]	Higher Order Logic (HOL)	Qualitative	Formal	Agnostic
[32]	Real valued functions	Continuous Dynamics	Formal	Agnostic
[19]	Stochastic petri nets	Stochastic	Formal	Agnostic
[12, 115]	Probabilistic timed automata	Stochastic, Time	Formal	Agnostic
[63, 65]	Timed Petri Nets	Time	Formal	Agnostic
[10]	Petri nets	Qualitative	Formal	Agnostic
[10]	UML class diagrams	Qualitative	Informal	Agnostic
[110, 162, 163]	Coulored Petri Nets	Qualitative	Formal	Agnostic
[77]	CEFSM - Communicating extended finite state machines	Qualitative	Formal	Agnostic
[108, 109]	Stackelberg policies	Arithmetic	Informal	Agnostic
[39–41, 130, 154, 175, 197]	Communicating Sequential Processes (CSP)	Quantitative	Formal	Agnostic
[201]	Roadview models	Continuous Dynamics	Informal	Traffic
[164]	Computation Graph	Qualitative	Formal	Agnostic
[84, 191, 192]	Promela	Qualitative	Formal	Agnostic
[104]	Knowledge Association Graph	Continuous Dynamics	Formal	Agnostic
[39–41]	RoboChart	Time, Stochastic	Formal	Robotics
[41]	RoboSim	Time, Stochastic	Formal	Robotics
[172]	Kripke model	Qualitative	Formal	Agnostic
[148]	DSLs	Qualitative	Informal	Agnostic
[171]	DSL for traffic scenario	Dynamics	Informal	Traffic
[198]	DSL for cognitive models	Qualitative	Informal	rogots
[101]	Open Interpreter Systems	Qualitative	Formal	Agnostic
[101]	Global Transition Systems	Qualitative	Formal	Agnostic
[45]	Geometrical model	Geometry	Informal	Robots
[1]	Functional Mockup Unit model	Dynamics	Informal	Agnostic
[1]	SystemC	Dynamics	Informal	Agnostic
[49]	Relational Functional Model	Qualitative	Formal	Agnostic

Continued on next page

Table 2 – *Continued from previous page*

Papers	Model	Model type	Formalism	Domain
[205]	Software Reliability Growth Model	Stochastic	Informal	Agnostic
[17]	Event-B	Qualitative	Formal	Agnostic
[50]	Open scenario framework	Arithmetic	Formal	Agnostic
[59]	DIME framework	Qualitative	Formal	Agnostic
[82]	Hybrid System model	Continuous Dynamics	Formal	CPS
[34, 35, 71]	Hybrid Automata	Continuous dynamics	Formal	CPS
[194]	Periodic Controlled Hybrid Automata	Continuous Dynamics	Formal	CPS
[195, 196]	Stochastic Hybrid Automata	Stochastic, Continuous dynamics	Formal	CPS
[106]	Safety Automata	Stochastic	Formal	Agnostic
[113]	Probabilistic agent templates	Stochastic	Formal	Agnostic
[176]	Finite State Abstract Model	Qualitative	Formal	Agnostic
[178]	Ontology model	Qualitative	Formal	Agnostic
[119, 120]	Process Algebra for Robot Schema (VIPARS)	Stochastic	Formal	Agnostic
[122]	Demand Compliant Design (DeCoDe-bsaed model)	Qualitative	Formal	Agnostic
[139]	ALICA	Arithmetic	Formal	Multi-agents
[61, 87]	Gwendolen	Arithmetic	Formal	Multi-agents
[52]	ETHAN (Gwendolen extension)	Arithmetic	Formal	Multi-agents
[73]	Klaim	Continuous dynamics	Informal	Agnostic
[73]	Mobile stochastic logic (MOSL)	Stochastic	Informal	Agnostic
[136]	Tabular use cases	Qualitative	Informal	Agnostic
[140]	Configuration Network Language	Time	Informal	Agnostic
[189]	RoboticSpec	Time	Informal	Robots
[191, 192]	Brahms	Dynamics	Informal	Multi-agent
[200]	SysML	Qualitative	Formal	Agnostic

5.2 RQ2: Effect

In this section, we review two different types of measures: the first type of measures coded and reviewed in this section are those measures used for evaluating efficiency, effectiveness and coverage of the various testing interventions. The second types of measures are the measures of quality used in testing the subject system; by reusing the terminology we classify them under efficiency (i.e., concerning timing and resources) and effectiveness (i.e., concerning safety and quality) of the subject system.

5.2.1 Measures for Interventions. Table 3 provides an overview of our coding of these measures, classified into efficiency (testing time or resources), effectiveness (testing quality), and coverage (testing adequacy). It is remarkable that about a

third of the papers included for this survey used a measure of efficiency, effectiveness, or coverage to evaluate their results. This shows a significant gap in using well-defined measures to evaluate and compare various interventions.

It is also noteworthy that the interventions were measured against a vastly different range of measures. Apart from some very basic notions of efficiency (testing time, or state-space size) [13, 26, 46, 83, 90, 100, 123, 138, 143, 161, 164] and coverage (such as state and transition coverage) [10, 13, 14, 51, 77, 110, 163], most other notions are only used for a single intervention. This emphasises the need for coming up with domain-specific and more sophisticated notions of efficiency, effectiveness, and coverage that are used for benchmarking and comparing various interventions. Some exceptions that concern domain-specific measures are hypervolume (as a domain-specific measure for the searched space) and generational distance (as a measure of distance from optimal solutions) [25], cost of testing for autonomous vehicles in Euros per kilometre [31], feature interaction coverage [2, 29], situation coverage [121], and neuron- [180] and surprise adequacy [92] coverage.

Table 3. Classification of measures considered in testing interventions into efficiency (testing time or resources), effectiveness (testing quality), and coverage (testing adequacy)

Papers	Effectiveness	Efficiency	Coverage
[170]	Accuracy of the image recognition (failure rates)		
[25]	Hypervolume in fixed time (search-space coverage in time)	Generational distance in time (distance to Pareto optimal solutions in time)	Hypervolume
[62, 66, 76, 83, 90, 100, 114, 119, 120, 123, 138–140, 154, 161, 176]		Testing (est gen. and exc., model checking) time)	
[13]		Test case generation time	
[16, 28, 57, 60, 74, 96, 134, 143, 164]		Test execution and simulation time	
[26]		Reduced test case execution time	
[31]		Testing cost (€/ km)	
[2]	Feature interaction failure		
[5, 10, 15, 62, 66, 90, 101, 113, 154, 161, 172, 191, 192, 203]		State-space size	
[10, 13, 14, 51, 77, 110, 162, 163]			Structural coverage metrics (state, code, function, transition, path coverage)
[2, 29]			Feature interaction (e.g., pairwise and n-wise coverage)
[180]			Neuron coverage
[92]	Distance-based surprise adequacy		Surprise adequacy coverage
[129]	Number and probability of faulty scenarios generated		
[7, 18]	Reachability		
[46]		Search time	
[121]			Situation (graph) coverage
[168, 179]			Requirement
[93, 178]	Number of test cases		

Continued on next page

Table 3 – *Continued from previous page*

Papers	Effectiveness	Efficiency	Coverage
[147]	Number of failures		
[176]	Number of counter-examples		
[133]		Precision	
[133]			Diversity
[169]	Accuracy of the simulation		

5.2.2 Measures for Subject Systems. In this section, we review the measures of quality for the system under test that are used in various interventions, presented in Table 4. Unlike the previous section, there is more prevalence of domain specific measures; two commonly used measures are spatial distance from the intended trajectory (and variants thereof) [27, 33, 104, 105], collisions and obstacle avoidance [19, 25, 54, 108, 115, 118, 137, 182]. The remaining measures are sparsely used across many different interventions.

Table 4. Classification of measures of quality for the subject systems used in testing interventions.

Papers	Effectiveness	Efficiency
[137]		Number of collisions over time
[93]		Time to collision
[106]	Probability of time to collision	
[60, 148, 189]	Performance and safety properties	
[105]	Safety for human operators	
[12]	Satisfied performance properties wrt. number of robots	
[148]	Number of failures	
[33, 183]	Requirements satisfaction	
[27, 28, 33, 43, 67, 104, 105]	Spatial deviation of intended behaviour	
[86]	Endurance distance and stairs traversal of robots	
[170]	Accuracy of the image recognition	
[11, 16, 19, 25, 42, 47, 54, 108, 115, 118, 137, 182, 207]	Collisions & obstacle avoidance	
[182]	Stability	
[61, 144]		Resource utilisation (e.g.: CPU)
[144]		Network usage
[60, 115]		Fuel consumption
[109]		Constraint violation rate
[61]	Search depth	
[12]	Throughput	

Continued on next page

Table 4 – *Continued from previous page*

Papers	Effectiveness	Efficiency
[12]		Device utilisation
[12]		Response time
[28]		Training time
[63]	Schedulability	
[126, 132]	Positive and supportive interactions towards humans	
[134]		Latency
[6]		Idle time
[72, 132]		Task completion time
[160]	Anthropomorphism measure	
[185]	Number of hazards and risk reduction measures	Time for hazard identification and risk reduction
[205]		Median miles to next disengagement
[15, 119, 120, 132, 140, 195, 196, 203, 204]	Probability of mission success and failure	
[203]	Number of recharges	Battery life
[7, 195, 196]	Formal assertions (deadlock freedom, liveness)	
[50]	Criticality (complexity of scenario and dynamics)	
[65, 74, 105, 178]	Vehicle performance (acceleration, speed, position)	
[128]	Regret (Difference between rewards earned and achievable rewards)	
[179]	Severity of failure	

5.3 RQ3: Tooling

We gather and describe the tools that have been employed and introduced amongst included studies. We categorise tools as context and effect tools; a context tool is one that has been employed by the intervention but it is not a byproduct of its respective work. Effect tools, on the other hand, are the tools that have been developed by the academic community in our list of selected papers.

5.3.1 Context Tools. As shown in table 6, tools for simulation are amongst the most utilised; their usefulness comes from a less costly method of visualising whether the design and process are satisfactory. The middleware ROS [149] combined with the 3D simulator Gazebo [95] form the most popular tool for robotics simulation. Furthermore, Simulink [56], a graphical extension of MATLAB [127], is the most used tool for modelling and simulation of dynamic systems.

In the context of autonomous vehicles, traffic simulators such as SUMO [21] and SYNCHRO [85] have also been employed by included interventions, along with vehicle simulators such as CarMaker [38] and Autoware framework [89].

Moreover, tools for formal verification are also extensively used, with model checkers being the most prominent type. The statistical model checker, Prism [102], provides modelling and analysis of systems of stochastic nature modeled in markov chains or probabilistic automata. As for qualitative models, UPPAAL [103] offers formal verification for timed automata models that can be, however, extended to employ data types.

Table 5. Tools used in the context of testing interventions for RAS and description of tools.

Papers	Name	Description
[15, 18, 33, 113, 115, 131, 132, 147, 203, 204]	Prism	Probabilistic model checker
[55, 199]	Maude	LTL model checker
[1, 4, 25, 27, 29, 76, 82, 129, 168, 173, 177]	Matlab/Simulink	System design and simulation environment
[137]	Jadex	BDI reasoning engine
[11, 13, 14, 30, 37, 43, 44, 83, 105, 124, 156, 188, 193]	ROS	Robotics middleware
[123]	Acceleo	Code generator
[7, 7, 12, 14, 16, 63, 78, 87, 161, 195, 196]	UPPAAL	Timed automata model checker
[13, 14, 30, 37, 43, 44, 57, 69, 124, 183, 184, 193, 207]	Gazebo	Robots simulator
[79, 91, 141, 171, 177]	Unity-3D	3D games engine
[1, 2, 129]	PreScan	ADAS simulation platform
[125]	Alloy analyser	Analysis of Alloy models
[69, 75, 144, 177]	SUMO	Traffic modelling and simulation
[144]	UsarSim	Robot simulation
[54]	P	Robotic logics model checker
[54]	Breach	Cyber-Physical Systems simulation
[51]	PATeCa	Autonomous vehicles field testing analysis
[118]	JavaMoP	Runtime monitoring framework
[187]	Isabel/HOL	Automated theorem prover
[19]	Cosmos	Tool for statistical models
[64, 65]	Fiacre	Formal verification for TOPCASE environment
[64, 65]	TINA	Circuit simulator
[65]	Genom3	Robotic components development
[10]	Pipe	Petri Nets editor and analyser
[10, 106]	CADP	Design of communication protocols and concurrent systems
[36, 66, 90]	NuSMV	Extension of SMV symbolic model checker
[53, 61, 62, 87]	MCAPL framework	Prototyping and model checking BDI agents
[39, 40, 130, 154, 175, 197]	FDR	CSP model checker

Continued on next page

Table 5 – *Continued from previous page*

Papers	Name	Description
[164]	haROS	Static analysis of ROS application code
[84]	Spin	PROMELA model checker
[87]	Torcs	Car racing simulator
[172]	SMV	Symbolic model checker
[60]	Synchro	Traffic simulation and analysis
[60]	Cplex	Mathematical programming solver
[79]	Apollo	Autonomous driving platform
[67, 141]	TensorFlow	Machine learning platform
[173]	Simscape	Multidomain physical systems simulator
[1]	Veloce	Hardware simulator
[67]	DeepDriving	Vision-based learning for autonomous cars
[168]	Supremica	Modelling and analysis of discrete-event control functions
[168]	TLC Model Checker	TLA+ model checker
[1]	Simcenter Amesim	Analysis of digital systems
[11, 74]	IPG CarMaker	Autonomous vehicle simulator
[11]	Autoware	Simulation and analysis of self-driving vehicles
[39]	Wodel	Mutant generator
[185]	Zot	TRIO model checker
[188]	YOLO	Real-time object detection system
[188]	OpenSceneGraph (OSG)	3D graphics toolkit
[168]	Spark	Programming language and verification toolset
[59]	Gear	DIME model checker
[82]	Wolfram Mathematica	Computing system
[106]	CARLA	Autonomous vehicle simulator
[128]	Multi-Attribute Task Battery II (MATB-II)	NASA workload simulation tool
[119, 120, 140]	VIPARS	PARS model checker
[139]	Clingo	Programming solver
[71]	ARIADNE	Reachability analysis of hybrid automata
[192]	BrahmsToPromela	Translation tool from Brahms to PROMELA
[193]	Orocos toolchain	Creation of real-time robotics applications

5.3.2 Effect Tools. A total of 22 tools, publicly available or otherwise, have been introduced by the academic community as an effect of their intervention. Five of them were not accessible at the time of writing this survey and were classified as private, including SSIM [96], which is a tool for simulating flight software employed in Mars rovers projects. The

remaining tools, a total of 17, are available for the general public; 15 of those also have the source artefacts made public and have been classified as open-source.

Analogously to context tools, we notice a focus on development of tools for simulation and model checking. Tools for testing vehicles are amongst such tools, including in road [4, 47, 67, 138, 201], aerial [57, 117] and maritime [45] environments. As for robots, RoboTool [41] and Improv [6] offer formal verification alternatives for testing robots whilst ROSRV [83] provides a ROS extension for verification at runtime.

Table 6. Tools introduced by studies included in this survey for testing RAS.

Papers	Name	Description	Availability
[180]	DeepTest	Testing of DNN-driven autonomous cars	Open-source
[138]	APEX	Formal verification of autonomous vehicle trajectory planning	Private
[65]	Translation tool	Translation tool from GenoM to Fiacre	Private
[201]	Roadview	Traffic scene simulator for Autonomous Vehicles	Private
[39–41, 130]	RoboTool	Formal verification and simulation of robots	Public
[117]	MAV3DSim	Simulation platform for UAV controllers	Open-source
[4]	Florida Poly AV Verification Framework (FLPolyVF)	Verification of the decision making of autonomous vehicles	Open-source
[96]	Simulator in Julia	Robots simulation	Open-source
[45]	Stonefish	Simulation tool for marine robots	Open-source
[57]	GzUAV	Framework to run multiple-UAV simulations in Gazebo	Open-source
[47]	Move	Suite of tools to test autonomous vehicles	Open-source
[184]	SSIM	Simulation of flight software	Private
[6]	IMPROV	Tool for self-verification of robots	Open-source
[16]	VerifCar	Framework for validation of decision policies of communicating autonomous vehicles	Open-source
[18]	MCpMC	Statistical model checking of pMC	Open-source
[134]	Asynchronous Multi-Body Framework	Simulation of multi-body systems	Open-source
[46]	RobTest	Tool for stress testing of Single-arm robots	Private
[67]	AsFault	Test case generation for self-driving cars	Open-source
[202]	CyberEarth	Simulation of robots and cyber-physical systems	Public
[33]	Argos	Multi-physics robot simulator	Open-source
[54]	Drona	Programming framework for robotic systems	Open-source
[83]	ROSRV	Runtime verification framework for ROS	Open-source
[69]	Hybrid Simulation	3D simulation tool	Open-source
[76]	Spot	Prediction of traffic participants	Open-source

Continued on next page

Table 6 – *Continued from previous page*

Papers	Name	Description	Availability
[82]	FROST*	Modelling and simulation of dynamical systems	Open-source
[113]	PSV-CA	Probabilistic swarms verifier	Open-source
[147]	RoVer	Model Checker	Open-source
[80]	Formal	Modelling and symbolic execution of CPS	Private
[124]	UUV	Gazebo extension for underwater scenarios	Open-source
[156]	V-REP	Robots simulator	Open-source
[181]	MARS	Simulation environment for marine swarm robotics	Open-source
[66]	Cruton	Translation from robotics DSL into NuSMV	Open-source
[133]	Range Adversarial Planning Tool (RAPT)	Test scenarios generation	Open-source
[167]	Pegasus	Autonomous vehicles simulation	Pegasus
[169]	AirSim	Drone simulation environment	Public
[193]	Cosina	Simulation of real-time robotics systems	Open-source
[114]	MCMAS	Multi-agent systems model checker	Open-source

5.4 RQ4: Applicability

Table 7 provides an overview of the case studies conducted amongst included papers. We classify them as small, benchmark, and industrial. Case studies designed specifically to evaluate a particular intervention, which lack sufficient details or generality to be employed for a general class of interventions, were classified as small. On the other hand, those cases studies that are sufficiently general and contain details to evaluate a range of interventions, provided that they are not used in an industrial context, are categorised as benchmark. Industrial case studies are those real-world (and hence, typically detailed and complex) cases conducted in a industrial setting.

Our observation identifies a significant gap in industrial evaluation of interventions; only seven interventions ([1, 2, 25, 60, 86, 168, 184]) have been evaluated in a industrial context. Understandably, the majority of cases studies have been fully conducted in academic settings. Of those, approximately half made use of small-scale models, which are often not representative of real systems. The other half, employed their proposed interventions on large scale subjects and data-sets, including physical systems.

Table 7. Classification of case studies considered in testing interventions as small, industrial, and benchmark.

Papers	Small	Industrial	Benchmark
[148]			Autonomous off-road robot RAVON
[33]			Swarm of robots
[70]	Pedestrian detection		

Continued on next page

Table 7 – *Continued from previous page*

Papers	Small	Industrial	Benchmark
[199]	Humanised robots		
[17, 18, 27, 39, 49, 52, 77, 111, 202]	UAV		
[137]	Cleaner agent		
[25]		ADAS System	
[105]			2 wheels differential drive robots
[123]	Self-driving vehicle		
[29]			UAV / Land vehicle cooperation
[182]			Smores
[180]			Udacity
[31]	Sensor system		
[13, 14]			BERT 2
[186]	Software functions		
[165]			Connected robots
[2]		Self-driving system	
[151]			MIT and NIRA datasets
[125]	Family of surgical robots		
[170]			Traffic sign database
[86]		Emergency response robot	
[144]			Benchmark
[51]			Carina I
[44]	Path planning and decision making		
[54]	Surveillance drone		
[69, 138]	Lane-changing scenarios		
[78, 164]			Kobuki robot
[118]			LEGO EV3 robot
[53, 55, 187]	Small robot		
[19]	Simple controllers		
[115]	Unmanned Surface Vehicles		
[65]			RMP400 Robot MANA
[10]	USAR robots		
[110]	Cooperative forklifts		
[161]	Agricultural robot		
[90, 108]	Cruise control		
[109]	Traffic environment		
[83]			Landshark
[197]	Mullti-agent manufacturing controller		

Continued on next page

Table 7 – *Continued from previous page*

Papers	Small	Industrial	Benchmark
[194]			Alice autonomous vehicle
[32]			Parallel delta robot
[37]	AR.Drone		
[84]	Cooperative UAVs		
[12]	(Industrial scale) transport robot		
[87]	Platoon		
[104]			Jack ROV
[135]			UAV
[7, 7, 40, 98]	Robot swarm		
[64]			Quadcopter controller
[143]	iCub robot		
[4, 42, 106]	Collision avoidance scenarios		
[28, 141]			Videos of pedestrians and vehicles
[47]			Traffic wave observations
[57]			Leader and follower UAVs
[60]		Test drive in a test track	
[63]			ROBNAV mobile robot
[92]			Udacity, MNIST and CIFAR-10 datasets
[96]			ATLAS robot
[101]	Trained gate controller		
[6, 126]			Human-robot interactions
[107, 129, 195, 196, 200]	Autonomous vehicles scenarios		
[134]			DaVinci research kits
[173]			Turtlebot 2
[177]			ZalaZone Smart City Zone
[41]	Footbot		
[184]		Mars Rover	
[1]		Automated breaking system	
[11]	Path following autonomous vehicle		
[43]	Autonomous parking		
[43]	Car following		
[46]	Single arm robot		
[160]	Ultimatum game		
[168]		Lateral State Manager	
[185]			Flexible Manufacturing System (FMS)

Continued on next page

Table 7 – *Continued from previous page*

Papers	Small	Industrial	Benchmark
[188]			Drone with Pixhawk flight controller
[205]			WAYMO public road testing dataset
[204]			Unmanned underwater vehicle (UUV)
[203]			Windfarm drone
[76]			Traffic Scenarios
[82]			ATLAS and DRC-HUBO robots
[74]		ADAS scenarios	
[147]			NAO robot
[93, 178]		Automated Emergency Break	
[128]		NASA benchmark and user case studies	
[175]	LEGO EV3 robot		
[176]	Simple robot with LiDAR		
[9]	Border control system		
[80]			NASA's Unmanned Ground Vehicle
[119]	Military overwatch missions		
[122]	"AMiResot" robot platform		
[124]		RexROV and Desistek SAGA mini-ROV	
[139]	Service robot		
[162]	Re-configurable autonomous trolley		
[181]			Hanse UAV
[183]		Cartesian impedance Control System in torque mode	
[15]			iRobot vaccum cleaner
[30]			KUKA LWR4+ and the Universal Robots UR5,
[34, 35]	Surgical robot		
[62]	Cruise control agent		
[66]		Care-O-bot	
[71]	Paint spray robot		
[73]	Group of robots		
[132]	Communicating robots		
[133]			Underwater vehicle
[140]	Search mission		

Continued on next page

Table 7 – *Continued from previous page*

Papers	Small	Industrial	Benchmark
[154]			Chemical detector
[157]		Farming	
[169]		Quadrotor with Pixhawk controller	
[189]			OUR-1 robot
[191, 192]			Care-O-bot
[193]			COMAN
[207]			CoCar parking
[206]		Adaptive cruise control	

6 SUGGESTIONS AND RECOMMENDATIONS TO STUDY AUDIENCE

In this section, we analyse the results of the previous sections in order to identify relative strengths and weaknesses regarding our research questions and for our two target audience groups: researchers and practitioners. We do this, by analysing the results in terms of sub-domains of RAS for which the interventions have been introduced, as well as, whether they consider the connected and cooperative aspects of RAS. We also review the different testing strategies used in the reviewed interventions. We conclude this section by drawing recommendations from our analysis both for researchers and practitioners.

6.1 Analysis

6.1.1 Domain. Table 8 provides a concise summary of the domains covered by the reviewed interventions. A bulk of reviewed interventions do not pertain to any specific sub-domain of RAS. This indicates a clear gap for subdomain-specific research that considers the characteristics of each of these subdomains and takes them into account in their testing interventions. Most importantly, the subdomain of testing marine and submarine RAS as well as space RAS is under-explored (the only included intervention regarding marine and submarine robots [45, 124, 146, 181] and regarding space robots [128, 184] are not represented in Table for the sake of brevity). We note that there is a recently funded European project REMARO to fill in this substantial gap[2].

Furthermore, in Table 8, we map the identified subdomains to the different aspects of our research questions as follows:

RQ1 Across all sub-domains a majority of models have been formal and quantitative and substantial gaps can be detected (most notably in the aerial vehicles sub-domain) regarding using qualitative and informal models for testing.

RQ2 Across all studied sub-domains, there is a clear gap in using precise notions of effectiveness, efficiency, and coverage. Among these, some generic notions of effectiveness and efficiency (such as testing time and state space size) and the notion of coverage (such as node and transition coverage) are the most-used measure for quantifying the effect. Common, more sophisticated measures of effectiveness, efficiency, and adequacy such as Average Percentage of Faults Detected (APFD) [158] do not seem to have been adopted in or extended to the

[2]https://cordis.europa.eu/project/id/956200

		Vehicles		Robots	
		Road	Aerial	Mobile	Generic / Non-Physical / Immobile
RQ1	Qualitative	[51, 52, 61, 62, 70, 80, 123, 136, 168, 170, 176, 178, 198, 200]	[17, 36, 49, 77]	[10, 125]	[8, 9, 14, 22–24, 53, 55, 59, 98–101, 110, 112, 114, 122, 137, 139, 159, 163, 164, 185, 191, 192, 197, 199]
	Quantitative	[1, 16, 19, 25, 27, 44, 47, 50, 60, 74, 87, 90, 106–109, 111, 129, 131, 138, 142, 144, 171, 177, 194–196, 201, 205]	[18, 29, 37, 54, 84, 117, 135, 172, 203]	[12, 33, 115, 148]	[6, 7, 15, 30, 34, 35, 39–41, 45, 46, 63–66, 71, 73, 78, 82, 83, 96, 104, 105, 113, 118, 119, 124, 130, 132, 140, 143, 147, 154, 156, 161, 162, 173, 175, 181, 184, 187, 189, 193, 204]
	Formal	[16, 19, 51, 52, 61, 62, 80, 87, 90, 106, 111, 131, 138, 142, 168, 170, 178, 194–196, 198, 200]	[17, 18, 29, 36, 49, 54, 77, 84, 172, 203]	[10, 12, 33, 115, 125, 148]	[7–9, 14, 15, 22–24, 34, 35, 39–41, 53, 55, 59, 63–66, 71, 78, 98–101, 104, 110, 112–114, 118, 119, 122, 130, 132, 137, 140, 143, 147, 154, 159, 161–164, 175, 185, 187, 189, 191, 192, 197, 199, 204]
	Informal	[1, 25, 27, 44, 47, 60, 70, 107–109, 123, 129, 136, 144, 171, 177, 201, 205]	[37, 57, 117, 135]		[6, 30, 45, 73, 82, 83, 96, 105, 124, 139, 156, 173, 181, 184, 193]
RQ2	Effectiveness	[2, 11, 16, 19, 25, 27, 28, 42, 43, 47, 50, 60, 61, 67, 92, 93, 106, 108, 129, 133, 168, 170, 178, 179, 195, 196, 207]	[18, 54, 169, 203]	[12, 33, 115, 148]	[7, 15, 63, 104, 105, 118, 119, 126, 128, 132, 137, 160, 182, 183, 185, 204]
	Efficiency	[16, 28, 31, 60–62, 74, 76, 90, 93, 109, 123, 133, 138, 144, 176, 205]	[57, 172, 203]	[10, 12]	[6, 15, 34, 35, 46, 66, 72, 83, 96, 101, 113, 114, 119, 132, 134, 143, 154, 161, 164, 185, 197]
	Coverage	[51, 92, 121, 179]	[29, 77]		[14, 110, 162, 163, 180]
RQ3	Open-source	[4, 16, 47, 67, 69, 76, 180]	[18, 57, 117]		[6, 33, 45, 54, 66, 82, 83, 96, 113, 114, 124, 134, 147, 156, 181, 193]
	Public	[133]	[169]		[41, 202]
	Proprietary	[80, 138, 167, 201]			[46, 65, 184]
RQ4	Small	[4, 11, 19, 26, 27, 31, 42–44, 52, 61, 62, 69, 70, 79, 87, 90, 106–109, 111, 123, 129, 138, 186, 195, 196, 200, 201]	[17, 18, 36, 37, 49, 54, 77, 84, 117, 172]	[10, 12, 115, 125]	[7, 9, 15, 22, 24, 34, 35, 39–41, 46, 53, 55, 59, 71–73, 98, 99, 101, 110, 113, 114, 122, 132, 137, 140, 143, 156, 159–163, 175, 181, 187, 197, 199, 202]
	Industrial	[1, 2, 25, 60, 74, 93, 167, 168, 178, 206]	[157, 169]	[157]	[66, 86, 124, 128, 166, 183, 184]
	Benchmarks	[28, 47, 51, 67, 76, 80, 92, 133, 141, 142, 144, 170, 177, 194, 205, 207]	[29, 57, 135, 188, 203]	[33, 148]	[6, 14, 30, 32, 63–65, 78, 82, 83, 96, 104, 105, 118, 126, 134, 147, 151, 154, 164, 165, 173, 180, 182, 185, 189, 191–193, 204]

Table 8. Testing, Validation, and Verification Interventions for Specific Subdomains of RAS

domain of RAS. We do see some recent trend towards domain-specific notions of effectiveness and coverage [2, 25, 29, 31, 92, 121, 180]; almost all of these notions have been applied to the autonomous vehicles domain, but most of them can be adapted to be applicable to other domains as well.

RQ3 There is a considerable gap concerning tool support for testing RAS. There are very few open source tools, mostly in the autonomous vehicles [4, 16, 47, 67, 69, 76, 180] and aerial vehicles [18, 57, 117] sub-domains. No open-source tools support the domain-specific aspects of mobile robotic system. The same pattern with a more sever gap is present for proprietary tools. Very few public (but not open source) tools are developed or used in the reviewed literature.

RQ4 There is also a very severe gap across all subdomains in using industrial case studies for evaluating RAS testing interventions. The most notable exceptions are a handful of case studies, mostly in the autonomous vehicles [1, 2, 25, 60, 74, 93, 167, 168, 178, 206] and the aerial vehicles [66, 86, 124, 128, 166, 183, 184] sub-domains, performed in an industrial context. Many interventions used small case studies, mostly without any specific application subdomain (e.g., using generic models of mobile robots); in these cases, the models did not contain enough details to be part of a general benchmark. There have also been some evaluations performed on small case studies based on drones and UAVs.

6.1.2 Cooperation and Connectivity. Table 9 provides an overview of the interventions used to test cooperation and connectivity in RAS. The interventions can be broadly categorised into swarm RAS, where an emerging behaviour is to be observed through cooperation of a large number of RAS, versus cooperative RAS where few RAS units engage in a well-defined interaction (possibly with their environment) to achieve a goal.

In general, this turns out to be an understudied area of testing RAS and little focus has been put in testing cooperative and connected aspects of RAS in the literature. For the very few interventions reported in the literature, there is scarcely any evidence of efficiency or effectiveness available. The handful of reported evaluations are only performed on small-scale case studied and are not accompanied by open source tools. In our analysis, we focused on cooperation among robots; however, only in 2019, we encountered some papers that study cooperation from a human-robot interaction viewpoint [6, 126, 160].

		Swarm	**Cooperative**
RQ1	**Qualitative**	[49, 98–100]	[55, 110, 163]
	Quantitative	[7, 33, 40, 113]	[16, 29, 84, 87, 132, 162]
	Formal	[7, 33, 40, 49, 98–100, 113]	[16, 29, 55, 84, 87, 110, 132, 162, 163]
	Informal		[57]
RQ2	**Effectiveness**	[7, 33]	[16, 132]
	Efficiency	[113]	[16, 57, 132]
	Coverage		[29, 110, 162, 163]
RQ3	**Open-source**	[33, 113]	[16, 57]
	Public	[41]	
	Private		
RQ4	**Small**	[7, 40, 49, 98, 99, 113]	[55, 84, 87, 110, 132, 146, 162, 163]
	Industrial		[157]
	Benchmarks	[33]	[29, 57]

Table 9. Testing Cooperation and Connectivity in RAS

RQ1 The number of interventions is in general very low to draw any meaningful strength. Qualitative aspects are not treated as much as quantitative aspects; verification starting from informal models is clearly understudied.

RQ2 Very little has been done to rigorously study the effect of the intervention in terms of effectiveness, efficiency and coverage for connected RAS. For Swarm RAS, we did not encounter any rigorous measure of effectiveness, efficiency, or coverage.

RQ3 There is clearly a lack of tools for testing connected RAS. In particular, we did not encounter any open source or public tools for this purpose within our scope.

RQ4 No effort has been spent on evaluating the interventions on industrial-scale and benchmarks. We believe coming up with and using benchmarks for evaluating such intervention may be of significant impact.

6.1.3 Testing strategy. Table 10 provides an overview of the testing strategies used for RAS. By far the most widely used strategy is formal verification, followed by simulation and runtime monitoring, respectively. Model-based testing is the least researched strategy.

RQ1 By far quantitative testing techniques are the most widely researched strategies (this was also a common observation for the domain and connectivity aspects).

RQ2 Among the measures used for evaluating interventions efficiency is most often used. There is virtually no intervention that is evaluated using a notion of coverage; the only exception is [10], where a notion of simple-path coverage is used to measure adequacy for a verification technique based on Petri nets. It is notable that for runtime monitoring only in one publication [118], a notion of efficiency is used.

RQ3 There is a considerable lack of tools for simulation, model-based testing and runtime monitoring. For formal verification, there seems to b some considerable strength in terms of tool support.

RQ4 Nearly all interventions used small-scale case studies for their evaluations. Notable exceptions are applications to autonomous vehicles [190, 194] and autonomous transport robots [12].

6.2 For researchers

Throughout the various categories we have coded in this case study, the most prominent gap is in the use of agreed-upon rigorous measures to evaluate the efficiency and effectiveness of the interventions as well as real-world benchmarks that can be used to evaluate such measures. As observed in the earlier sections, much of the measures of efficiency and effectiveness measures are very generic and there is also a relative gap in domain-specific measures suitable for the RAS sub-domains.

Another considerable gap is in the use of quantitative specification languages to specify the desired properties of the system; due to the inherent heterogeneity of RAS, we need to have property languages that cover aspects such as the combination of discrete and continuous dynamics as well as stochastic and epistemic aspects that may be used to model the aspects of behaviour concerning the environment and the users. Connected to this point is the relative gap in interventions that perform a quantitative analysis of the system and provide quantitative metrics of quality as the outcome of the test. Some starting points in this direction are the use of quantitative properties that incorporate probabilistic and stochastic- [33, 143], timed- [78, 138, 172], and continuous dynamical [54] aspects of RAS. We have also noted the use of a specification language that caters for a combination of stochastic and continuous aspect of RAS [115]. On the contrary there is a relative strength in using qualitative models, including property specification languages as predicate- [8] and temporal logics [19, 36, 65, 84, 87, 118, 142, 174], as well as epistemic extensions thereof [100, 112]. Also, there is a wealth of studies on the use of discrete relational [125], state-based [33, 40, 77, 118, 130, 197, 199] and

		Simulation	Model Based Testing	Formal Verification	Runtime Monitoring
RQ1	**Qualitative**	[49, 80, 164]	[10, 14, 77, 110, 122, 137, 163]	[8, 9, 17, 22–24, 36, 52, 53, 55, 59, 61, 62, 98–101, 112, 114, 123, 139, 159, 168, 176, 185, 191, 192, 197, 199]	
	Quantitative	[1, 33, 37, 41, 45, 47, 73, 82, 96, 104, 107–109, 117, 124, 129, 130, 135, 144, 156, 171, 177, 181, 184, 193, 201]	[25, 27, 46, 111, 148, 162]	[6, 12, 16, 18, 19, 29, 33–35, 39–41, 50, 54, 63–66, 71, 73, 78, 84, 87, 90, 113, 115, 119, 130, 132, 138, 140, 142, 143, 147, 154, 161, 172, 175, 187, 194–196, 203, 204]	[54, 74, 83, 118, 142, 173, 189]
RQ2	**Effectiveness**	[11, 28, 33, 42, 43, 47, 67, 104, 108, 129, 169, 182, 183]	[25, 27, 133, 148]	[12, 16, 18, 19, 33, 50, 54, 61, 63, 115, 119, 132, 168, 185, 195, 196, 203, 204]	[54, 118]
	Efficiency	[31, 57, 72, 96, 109, 134, 144, 164]	[10, 28, 46, 133]	[6, 12, 16, 34, 35, 61, 62, 66, 76, 90, 101, 113, 114, 119, 123, 132, 138, 143, 154, 161, 172, 176, 185, 197, 203]	[74, 83]
	Coverage		[14, 77, 110, 162, 163]	[29]	[121]
RQ3	**Open-source**	[6, 33, 45, 47, 57, 67, 69, 82, 83, 96, 117, 124, 134, 156, 181, 193]		[4, 6, 16, 18, 54, 66, 76, 113, 114, 147]	[54, 83]
	Public	[41, 169, 202]	[133]	[41]	
	Private	[80, 167, 184, 201]	[46]	[65, 138]	
RQ4	**Small**	[11, 31, 37, 41–43, 49, 72, 73, 79, 107–109, 117, 129, 156, 181, 201, 202]	[10, 27, 46, 77, 110, 111, 122, 137, 162, 163]	[9, 12, 17–19, 22, 24, 34–36, 39–41, 52–55, 59, 61, 62, 71, 73, 84, 87, 90, 98, 99, 101, 113–115, 123, 132, 138, 140, 143, 159, 161, 172, 175, 187, 195–197, 199]	[54]
	Industrial	[1, 124, 157, 167, 169, 183, 184, 206]	[25]	[66, 168]	[74]
	Benchmarks	[28, 32, 33, 47, 57, 67, 80, 82, 96, 104, 134, 135, 141, 144, 164, 173, 177, 182, 188, 193]	[14, 133, 148]	[6, 29, 33, 63–65, 76, 78, 142, 147, 154, 185, 191, 192, 194, 203, 204]	[83, 118, 142, 189]

Table 10. Overview of the testing strategies used for RAS.

belief-based [13, 14, 87, 137] abstract models in testing and verification of RAS. Also several studies used informal simulation models for simulation tools such as Gazebo and USARSim [13, 14, 37, 37, 44, 83, 105, 144]. A suitable middle-ground may be the semi-formal and domain-specific models such as those built in Matlab/Simulink [25, 27, 135].

Regarding techniques, most of the techniques used so far in the literature have been formal verification techniques applied (on relatively high-level) qualitative [8, 36, 61, 100, 112, 123, 159, 197, 199] or quantitative [12, 19, 29, 33, 40, 54, 64, 65, 78, 84, 87, 90, 115, 130, 138, 142, 143, 161, 172, 187, 194] models of RAS. There is also some strength in the use of informal simulation techniques [33, 37, 104, 108, 109, 117, 130, 135, 144, 164, 201]. We have seen relatively few model-based testing [10, 14, 25, 27, 77, 110, 137, 148, 163] and run-time verification [54, 83, 118, 142] techniques that have been applied to (models of) complex and detailed RAS. We hence see a gap, and a trend towards closing this gap, in dynamic and non-exhaustive testing of RAS techniques.

Finally, lack of public tooling and data-sets is a major gap observed in the literature. They are very few techniques that are accompanied by a tool and there are very few public tools and data-sets for testing RAS [40, 61, 65, 83, 100, 117, 130, 187, 201].

6.3 For practitioners

The most significant gap is lack of industrial evaluation of existing interventions. There have been less than a handful of interventions applied in an industrial context and to systems of industrial complexity [2, 25, 86].

The human- and information-source is another aspect of testing interventions that is a severely understudied. We note a recent trend in combining user studies (in the sense of human-computer- and human-robot interactions) and traditional testing, validation, and verification techniques [6, 126, 160].

Also there is a gap in defining and evaluating testing processes, particularly in industrial contexts.

The lack of industrial- and domain-expert input into the models and techniques is evident and has led to generic and relatively simple modelling techniques and property languages being used for most intervention. Co-production with industrial partners can enrich these aspects and lead to models that can deal with the heterogeneity and complexity of industrial RAS.

7 CONCLUSION

We performed a systematic review of the interventions for testing robotics and autonomous systems in order to answer the following research questions:

(1) What are the *types of models* used for testing RAS?
(2) Which *efficiency and effectiveness measures* were introduced or used to evaluate RAS testing interventions?
(3) What are the interventions supported by *(publicly available) tools* in this domain?
(4) Which interventions have *evidence of applicability* to large-scale and industrial systems?

To this end, we started off by performing a pilot study on a seed of 26 papers. Using this pilot study, we designed and validated a search query, designed rigorous inclusion and exclusion criteria and developed an adaptation of the SERP-Test taxonomy. Subsequently, we went through two phases of search, validation and coding, in total going through a total of 10,534 papers. We finally coded the set of 202 papers and analysed them to answer our research questions.

A summary of the findings of the review with regards to our research questions is provided below:

(1) There is a wealth of formal and informal models used for testing RAS. In particular, there is a sizeable literature on using generic property specification languages (such as linear temporal logic) and qualitative modelling languages, such as variants of state machines, UML diagrams, Petri nets and process algebras. There is a clear gap in quantitative modelling languages that can capture the complex and heterogeneous nature of RAS. There is also a lack of domain-specific languages that can capture domain knowledge for various sub-domains of RAS.

(2) We observed a gap in rigorous and widely accepted metrics to measure effectiveness and efficiency, and adequacy of testing interventions. Similar to the previous items, those measures used in the literature are very generic and do not pertain to the domain specific aspects of RAS. Hence, there is a gap and a research opportunity for defining and evaluating rigorous (domain-specific) measures for efficiency, effectiveness, and adequacy for RAS testing interventions.
(3) There is a considerable number of intervention that rely on public tools to implement or evaluate their interventions. However, there are very few which make their proposed / evaluated interventions available for public use in terms of publicly available tools. There is hence a considerable gap in providing data-sets and public tools for further development of the field.
(4) There are less than a handful of testing interventions that have been evaluated in an industrial context. There have been some other interventions that used some real robots or autonomous systems, but in an academic context. This signifies the importance of future co-production between academia and industry in industrial evaluation of testing interventions for RAS.

ACKNOWLEDGMENTS

We would like to thank Jan Tretmans and Wojciech Mostowski for comments and discussions at the early stage of this research. Hugo Araujo and Mohammad Reza Mousavi have been partially supported by the UKRI Trustworthy Autonomous Systems Node in Verifiability, Grant Award Reference EP/V026801/1.

REFERENCES

[1] Mohamed AbdElSalam, Keroles Khalil, John Stickley, Ashraf Salem, and Bruno Loye. 2019. Verification of advanced driver assistance systems (ADAS) and autonomous vehicles with hardware emulation-in-the-loop. *International journal of automotive engineering* 10, 2 (2019), 197–204.
[2] Raja Ben Abdessalem, Annibale Panichella, Shiva Nejati, Lionel C Briand, and Thomas Stifter. 2018. Testing autonomous cars for feature interaction failures using many-objective search. In *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, Montpellier, France, 143–154.
[3] Nauman Bin Ali, Emelie Engström, Masoumeh Taromirad, Mohammad Reza Mousavi, Nasir Mehmood Minhas, Daniel Helgesson, Sebastian Kunze, and Mahsa Varshosaz. 2019. On the search for industry-relevant regression testing research. *Empir. Softw. Eng.* 24, 4 (2019), 2020–2055. https://doi.org/10.1007/s10664-018-9670-1
[4] Ala Jamil Alnaser, Mustafa Ilhan Akbas, Arman Sargolzaei, and Rahul Razdan. 2019. Autonomous vehicles scenario testing framework and model of computation. *SAE International Journal of Connected and Automated Vehicles* 2, 4 (2019).
[5] Matthias Althoff and John M Dolan. 2011. Set-based computation of vehicle behaviors for the online verification of autonomous vehicles. In *2011 14th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1162–1167.
[6] Matthias Althoff, Andrea Giusti, Stefan B Liu, and Aaron Pereira. 2019. Effortless creation of safe robots from modules through self-programming and self-verification. *Science Robotics* 4, 31 (2019).
[7] Saifullah Amin, Adnan Elahi, Kashif Saghar, and Faran Mehmood. 2017. Formal modelling and verification approach for improving probabilistic behaviour of robot swarms. In *2017 14th International Bhurban Conference on Applied Sciences and Technology (IBCAST)*. IEEE, 392–400.
[8] Benjamin Aminof, Aniello Murano, Sasha Rubin, and Florian Zuleger. 2015. Verification of asynchronous mobile-robots in partially-known environments. In *International Conference on Principles and Practice of Multi-Agent Systems*. Springer, 185–200.
[9] Benjamin Aminof, Aniello Murano, Sasha Rubin, and Florian Zuleger. 2016. Automatic verification of multi-agent systems in parameterised grid-environments. In *Proceedings of the 2016 international conference on autonomous agents & multiagent systems*. 1190–1199.
[10] Anneliese Andrews, Mahmoud Abdelgawad, and Ahmed Gario. 2016. World model for testing urban search and rescue (USAR) robots using petri nets. In *2016 4th International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*. IEEE, 663–670.
[11] Vimal Rau Aparow, Apratim Choudary, Giridharan Kulandaivelu, Thomas Webster, Justin Dauwels, and Niels de Boer. 2019. A Comprehensive Simulation Platform for Testing Autonomous Vehicles in 3D Virtual Environment. In *2019 IEEE 5th International Conference on Mechatronics System and Robots (ICMSR)*. IEEE, 115–119.
[12] Ryota Arai and Holger Schlingloff. 2017. Model-based performance prediction by statistical model checking an industrial case study of autonomous transport robots. In *Proc. 25th CS&P 2017-Concurrency, Specification and Programming*.

[13] Dejanira Araiza-Illan, Anthony G Pipe, and Kerstin Eder. 2016. Intelligent agent-based stimulation for testing robotic software in human-robot interactions. In *Proceedings of the 3rd Workshop on Model-Driven Robot Software Engineering*. 9–16.

[14] Dejanira Araiza-Illan, Tony Pipe, and Kerstin Eder. 2016. Model-Based Testing, Using Belief-Desire-Intentions Agents, of Control Code for Robots in Collaborative Human-Robot Interactions. *arXiv preprint arXiv:1603.00656* (2016).

[15] Rafael Araújo, Alexandre Mota, and Sidney Nogueira. 2017. Analyzing Cleaning Robots Using Probabilistic Model Checking. In *International Conference on Information Reuse and Integration*. Springer, 23–51.

[16] Johan Arcile, Raymond Devillers, and Hanna Klaudel. 2019. VerifCar: a framework for modeling and model checking communicating autonomous vehicles. *Autonomous agents and multi-agent systems* 33, 3 (2019), 353–381.

[17] Radu F Babiceanu and Remzi Seker. 2017. Formal verification of trustworthiness requirements for Small Unmanned Aerial Systems. In *2017 Integrated Communications, Navigation and Surveillance Conference (ICNS)*. IEEE, 6A3–1.

[18] Ran Bao, Christian Attiogbe, Benoit Delahaye, Paulin Fournier, and Didier Lime. 2019. Parametric statistical model checking of UAV flight plan. In *International Conference on Formal Techniques for Distributed Objects, Components, and Systems*. Springer, 57–74.

[19] Benoît Barbot, Béatrice Bérard, Yann Duplouy, and Serge Haddad. 2017. Statistical model-checking for autonomous vehicle safety validation.

[20] Halil Beglerovic, Steffen Metzner, and Martin Horn. 2018. Challenges for the Validation and Testing of Automated Driving Functions. https://doi.org/10.1007/978-3-319-66972-4_15

[21] Michael Behrisch, Laura Bieker, Jakob Erdmann, and Daniel Krajzewicz. 2011. SUMO–simulation of urban mobility: an overview. In *Proceedings of SIMUL 2011, The Third International Conference on Advances in System Simulation*. ThinkMind.

[22] Francesco Belardinelli, Panagiotis Kouvaros, and Alessio Lomuscio. 2017. Parameterised Verification of Data-aware Multi-Agent Systems.. In *IJCAI*. 98–104.

[23] Francesco Belardinelli, Alessio Lomuscio, Aniello Murano, and Sasha Rubin. 2017. Verification of Broadcasting Multi-Agent Systems against an Epistemic Strategy Logic.. In *IJCAI*, Vol. 17. 91–97.

[24] Francesco Belardinelli, Alessio Lomuscio, Aniello Murano, and Sasha Rubin. 2017. Verification of Multi-agent Systems with Imperfect Information and Public Actions.. In *AAMAS*, Vol. 17. 1268–1276.

[25] Raja Ben Abdessalem, Shiva Nejati, Lionel C Briand, and Thomas Stifter. 2016. Testing advanced driver assistance systems using multi-objective search and neural networks. In *Proceedings of the 31st IEEE/ACM International Conference on Automated Software Engineering*. 63–74.

[26] Christian Berger. 2015. Accelerating Regression Testing for Scaled Self-Driving Cars with Lightweight Virtualization–A Case Study. In *2015 IEEE/ACM 1st International Workshop on Software Engineering for Smart Cyber-Physical Systems*. IEEE, 2–7.

[27] Kevin M Betts and Mikel D Petty. 2016. Automated search-based robustness testing for autonomous vehicle software. *Modelling and Simulation in Engineering* 2016 (2016).

[28] Huikun Bi, Tianlu Mao, Zhaoqi Wang, and Zhigang Deng. 2019. A Deep Learning-based Framework for Intersectional Traffic Simulation and Editing. *IEEE Transactions on Visualization and Computer Graphics* (2019).

[29] Janis BICEVSKIS, Artis GAUJENS, and Janis KALNINS. 2013. Testing of RUAV and UGV Robots' Collaboration in the Simulink Environment. *Baltic Journal of Modern Computing* (2013).

[30] Andreas Bihlmaier and Heinz Wörn. 2014. Robot unit testing. In *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*. Springer, 255–266.

[31] Eckard Böde, Matthias Büker, Ulrich Eberle, Martin Fränzle, Sebastian Gerwinn, and Birte Kramer. 2018. Efficient splitting of test and simulation cases for the verification of highly automated driving functions. In *International Conference on Computer Safety, Reliability, and Security*. Springer, 139–153.

[32] Sebastian Bohlmann, Volkhard Klinger, and Helena Szczerbicka. 2017. Integration of a physical system, machine learning, simulation, validation and control systems towards symbiotic model engineering. In *Proceedings of the Symposium on Modeling and Simulation of Complexity in Intelligent, Adaptive and Autonomous Systems*. 1–12.

[33] Manuele Brambilla, Arne Brutschy, Marco Dorigo, and Mauro Birattari. 2014. Property-driven design for robot swarms: A design method based on prescriptive modeling and model checking. *ACM Transactions on Autonomous and Adaptive Systems (TAAS)* 9, 4 (2014), 1–28.

[34] Davide Bresolin, Luca Geretti, Riccardo Muradore, Paolo Fiorini, and Tiziano Villa. 2015. Formal verification applied to robotic surgery. In *Coordination Control of Distributed Systems*. Springer, 347–355.

[35] Davide Bresolin, Luca Geretti, Riccardo Muradore, Paolo Fiorini, and Tiziano Villa. 2015. Formal verification of robotic surgery tasks by reachability analysis. *Microprocessors and Microsystems* 39, 8 (2015), 836–842.

[36] Julien Brunel and Jacques Cazin. 2012. Formal verification of a safety argumentation and application to a complex UAV system. In *International Conference on Computer Safety, Reliability, and Security*. Springer, 307–318.

[37] Qing Bu, Fuhua Wan, Zhen Xie, Qinhu Ren, Jianhua Zhang, and Sheng Liu. 2015. General simulation platform for vision based UAV testing. In *2015 IEEE International Conference on Information and Automation*. IEEE, 2512–2516.

[38] IPG CarMaker. 2014. Users guide version 4.5. 2. *IPG Automotive, Karlsruhe, Germany* (2014).

[39] Ana Cavalcanti, James Baxter, Robert M Hierons, and Raluca Lefticaru. 2019. Testing Robots Using CSP. In *International Conference on Tests and Proofs*. Springer, 21–38.

[40] Ana Cavalcanti, Alvaro Miyazawa, Augusto Sampaio, Wei Li, Pedro Ribeiro, and Jon Timmis. 2018. Modelling and verification for swarm robotics. In *International Conference on Integrated Formal Methods*. Springer, 1–19.

[41] Ana Cavalcanti, Augusto Sampaio, Alvaro Miyazawa, Pedro Ribeiro, Madiel Conserva Filho, André Didier, Wei Li, and Jon Timmis. 2019. Verified simulation for robotics. *Science of Computer Programming* 174 (2019), 1–37.

[42] Qianwen Chao, Xiaogang Jin, Hen-Wei Huang, Shaohui Foong, Lap-Fai Yu, and Sai-Kit Yeung. 2019. Force-based heterogeneous traffic simulation for autonomous vehicle testing. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 8298–8304.

[43] Shitao Chen, Yu Chen, Songyi Zhang, and Nanning Zheng. 2019. A novel integrated simulation and testing platform for self-driving cars with hardware in the loop. *IEEE Transactions on Intelligent Vehicles* 4, 3 (2019), 425–436.

[44] Yu Chen, Shitao Chen, Tangyike Zhang, Songyi Zhang, and Nanning Zheng. 2018. Autonomous vehicle testing and validation platform: Integrated simulation system with hardware in the loop. In *2018 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 949–956.

[45] Patryk Cieślak. 2019. Stonefish: An Advanced Open-Source Simulation Tool Designed for Marine Robotics, With a ROS Interface. In *OCEANS 2019-Marseille*. IEEE, 1–6.

[46] Mathieu Collet, Arnaud Gotlieb, Nadjib Lazaar, and Morten Mossige. 2019. Stress testing of single-arm robots through constraint-based generation of continuous trajectories. In *2019 IEEE International Conference On Artificial Intelligence Testing (AITest)*. IEEE, 121–128.

[47] Marc Compere, Garrett Holden, Otto Legon, and Roberto Martinez Cruz. 2019. MoVE: A Mobility Virtual Environment for Autonomous Vehicle Testing. In *ASME International Mechanical Engineering Congress and Exposition*, Vol. 59414. American Society of Mechanical Engineers, V004T05A097.

[48] A. Cortesi, P. Ferrara, and N. Chaki. 2013. Static analysis techniques for robotics software verification. In *IEEE ISR 2013*. 1–6.

[49] Piotr Cybulski. 2019. A Framework for Autonomous UAV Swarm Behavior Simulation. In *2019 Federated Conference on Computer Science and Information Systems (FedCSIS)*. IEEE, 471–478.

[50] Werner Damm and Roland Galbas. 2018. Exploiting learning and scenario-based specification languages for the verification and validation of highly automated driving. In *2018 IEEE/ACM 1st International Workshop on Software Engineering for AI in Autonomous Systems (SEFAIAS)*. IEEE, 39–46.

[51] Vânia de Oliveira Neves, Márcio Eduardo Delamaro, and Paulo Cesar Masiero. [n. d.]. Automated Structural Software Testing of Autonomous Vehicles. ([n. d.]).

[52] Louise Dennis, Michael Fisher, Marija Slavkovik, and Matt Webster. 2016. Formal verification of ethical choices in autonomous systems. *Robotics and Autonomous Systems* 77 (2016), 1–14.

[53] Louise A Dennis, Michael Fisher, and Alan FT Winfield. 2015. Towards verifiably ethical robot behaviour. *arXiv preprint arXiv:1504.03592* (2015).

[54] Ankush Desai, Tommaso Dreossi, and Sanjit A. Seshia. 2017. Combining Model Checking and Runtime Verification for Safe Robotics. In *Runtime Verification*, Shuvendu Lahiri and Giles Reger (Eds.). Springer International Publishing, Cham, 172–189.

[55] Ha Thi Thu Doan, François Bonnet, and Kazuhiro Ogata. 2018. Model checking of robot gathering. In *21st International Conference on Principles of Distributed Systems (OPODIS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[56] Simulink Documentation. 2020. Simulation and Model-Based Design. https://www.mathworks.com/products/simulink.html

[57] Fabio D'Urso, Corrado Santoro, and Federico Fausto Santoro. 2019. An integrated framework for the realistic simulation of multi-UAV applications. *Computers & Electrical Engineering* 74 (2019), 196–209.

[58] Emelie Engström, Kai Petersen, Nauman bin Ali, and Elizabeth Bjarnason. 2017. SERP-test: a taxonomy for supporting industry–academia communication. *Software Quality Journal* 25 (2017), 1269–1305.

[59] Giuseppe Airò Farulla and Anna-Lena Lamprecht. 2017. Model checking of security properties: a case study on human-robot interaction processes. In *2017 12th International Conference on Design & Technology of Integrated Systems In Nanoscale Era (DTIS)*. IEEE, 1–6.

[60] S Alireza Fayazi, Ardalan Vahidi, and Andre Luckow. 2019. A Vehicle-in-the-Loop (VIL) verification of an all-autonomous intersection control scheme. *Transportation Research Part C: Emerging Technologies* 107 (2019), 193–210.

[61] Lucas ER Fernandes, Vinicius Custodio, Gleifer V Alves, and Michael Fisher. 2017. A rational agent controlling an autonomous vehicle: Implementation and formal verification. *arXiv preprint arXiv:1709.02557* (2017).

[62] Angelo Ferrando, Louise A Dennis, Davide Ancona, Michael Fisher, and Viviana Mascardi. 2018. Verifying and validating autonomous systems: Towards an integrated approach. In *International Conference on Runtime Verification*. Springer, 263–281.

[63] Mohammed Foughali. 2019. On Reconciling Schedulability Analysis and Model Checking in Robotics. In *International Conference on Model and Data Engineering*. Springer, 32–48.

[64] Mohammed Foughali, Bernard Berthomieu, Silvano Dal Zilio, Pierre-Emmanuel Hladik, Félix Ingrand, and Anthony Mallet. 2018. Formal verification of complex robotic systems on resource-constrained platforms. In *2018 IEEE/ACM 6th International FME Workshop on Formal Methods in Software Engineering (FormaliSE)*. IEEE, 2–9.

[65] Mohammed Foughali, Bernard Berthomieu, Silvano Dal Zilio, Félix Ingrand, and Anthony Mallet. 2016. Model checking real-time properties on the functional layer of autonomous robots. In *International Conference on Formal Engineering Methods*. Springer, 383–399.

[66] Paul Gainer, Clare Dixon, Kerstin Dautenhahn, Michael Fisher, Ullrich Hustadt, Joe Saunders, and Matt Webster. 2017. CRutoN: Automatic verification of a robotic assistant's behaviours. In *Critical Systems: Formal Methods and Automated Verification*. Springer, 119–133.

[67] Alessio Gambi, Marc Mueller, and Gordon Fraser. 2019. Automatically testing self-driving cars with search-based procedural content generation. In *Proceedings of the 28th ACM SIGSOFT International Symposium on Software Testing and Analysis*. 318–328.

[68] Shenjian Gao and Yanwen Tan. 2017. Paving the Way for Self-driving Cars - Software Testing for Safety-critical Systems Based on Machine Learning : A Systematic Mapping Study and a Survey.

[69] Mario Garzón and Anne Spalanzani. 2018. An hybrid simulation tool for autonomous cars in very high traffic scenarios. In *2018 15th International Conference on Control, Automation, Robotics and Vision (ICARCV)*. IEEE, 803–808.

[70] Lydia Gauerhof, Peter Munk, and Simon Burton. 2018. Structuring validation targets of a machine learning function applied to automated driving. In *International Conference on Computer Safety, Reliability, and Security*. Springer, 45–58.

[71] Luca Geretti, Riccardo Muradore, Davide Bresolin, Paolo Fiorini, and Tiziano Villa. 2017. Parametric formal verification: the robotic paint spraying case study. *IFAC-PapersOnLine* 50, 1 (2017), 9248–9253.

[72] Achim Gerstenberg and Martin Steinert. 2019. Evaluating and Optimizing Chaotically Behaving Mobile Robots with a Deterministic Simulation. *Procedia CIRP* 84 (2019), 219–224.

[73] Edmond Gjondrekaj, Michele Loreti, Rosario Pugliese, Francesco Tiezzi, Carlo Pinciroli, Manuele Brambilla, Mauro Birattari, and Marco Dorigo. 2012. Towards a formal verification methodology for collective robotic systems. In *International Conference on Formal Engineering Methods*. Springer, 54–70.

[74] Christoph Gladisch, Thomas Heinz, Christian Heinzemann, Jens Oehlerking, Anne von Vietinghoff, and Tim Pfitzer. 2019. Experience paper: search-based testing in automated driving control applications. In *2019 34th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 26–37.

[75] João SV Gonçalves, João Jacob, Rosaldo JF Rossetti, António Coelho, and Rui Rodrigues. 2015. An integrated framework for mobile-based ADAS simulation. In *Modeling Mobility with Open Data*. Springer, 171–186.

[76] Felix Gruber and Matthias Althoff. 2018. Anytime safety verification of autonomous vehicles. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1708–1714.

[77] Seana Hagerman, Anneliese Andrews, and Stephen Oakes. 2016. Security testing of an unmanned aerial vehicle (UAV). In *2016 Cybersecurity Symposium (CYBERSEC)*. IEEE, 26–31.

[78] Raju Halder, José Proença, Nuno Macedo, and André Santos. 2017. Formal verification of ROS-based robotic applications using timed-automata. In *2017 IEEE/ACM 5th International FME Workshop on Formal Methods in Software Engineering (FormaliSE)*. IEEE, 44–50.

[79] Jani Erik Heikkinen, Salimzhan Gafurov, Sergey Kopylov, Tatiana Minav, Sergey Grebennikov, and Artur Kurbanov. 2019. Hardware-in-the-Loop Platform for Testing Autonomous Vehicle Control Algorithms. In *2019 12th International Conference on Developments in eSystems Engineering (DeSE)*. IEEE, 906–911.

[80] Constance L Heitmeyer and Elizabeth I Leonard. 2015. Obtaining trust in autonomous systems: tools for formal model synthesis and validation. In *2015 IEEE/ACM 3rd FME Workshop on Formal Methods in Software Engineering*. IEEE, 54–60.

[81] Philipp Helle, Wladimir Schamai, and Carsten Strobel. 2016. Testing of Autonomous Systems – Challenges and Current State-of-the-Art. *INCOSE International Symposium* 26, 1 (2016), 571–584. https://doi.org/10.1002/j.2334-5837.2016.00179.x

[82] Ayonga Hereid and Aaron D Ames. 2017. FROST*: Fast robot optimization and simulation toolkit. In *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 719–726.

[83] Jeff Huang, Cansu Erdogan, Yi Zhang, Brandon Moore, Qingzhou Luo, Aravind Sundaresan, and Grigore Rosu. 2014. ROSRV: Runtime verification for robots. In *International Conference on Runtime Verification*. Springer, 247–254.

[84] Laura R Humphrey. 2013. Model checking for verification in UAV cooperative control applications. In *Recent Advances in Research on Unmanned Aerial Vehicles*. Springer, 69–117.

[85] David Husch and John Albeck. 2004. Trafficware SYNCHRO 6 user guide. *TrafficWare, Albany, California* 11 (2004).

[86] Adam Jacoff, Hui-Min Huang, Elena Messina, Ann Virts, and Anthony Downs. 2010. Comprehensive standard test suites for the performance evaluation of mobile robots. In *Proceedings of the 10th Performance Metrics for Intelligent Systems Workshop*. 161–168.

[87] Maryam Kamali, Louise A Dennis, Owen McAree, Michael Fisher, and Sandor M Veres. 2017. Formal verification of autonomous vehicle platooning. *Science of computer programming* 148 (2017), 88–106.

[88] Y. Kang, H. Yin, and C. Berger. 2019. Test Your Self-Driving Algorithm: An Overview of Publicly Available Driving Datasets and Virtual Testing Environments. *IEEE Transactions on Intelligent Vehicles* 4, 2 (2019), 171–185.

[89] Shinpei Kato, Shota Tokunaga, Yuya Maruyama, Seiya Maeda, Manato Hirabayashi, Yuki Kitsukawa, Abraham Monrroy, Tomohito Ando, Yusuke Fujii, and Takuya Azumi. 2018. Autoware on board: Enabling autonomous vehicles with embedded systems. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*. IEEE, 287–296.

[90] Hojat Khosrowjerdi and Karl Meinke. 2018. Learning-based testing for autonomous systems using spatial and temporal requirements. In *Proceedings of the 1st International Workshop on Machine Learning and Software Engineering in Symbiosis*. 6–15.

[91] Baekgyu Kim, Yusuke Kashiba, Siyuan Dai, and Shinichi Shiraishi. 2016. Testing autonomous vehicle software in the virtual prototyping environment. *IEEE Embedded Systems Letters* 9, 1 (2016), 5–8.

[92] Jinhan Kim, Robert Feldt, and Shin Yoo. 2019. Guiding deep learning system testing using surprise adequacy. In *2019 IEEE/ACM 41st International Conference on Software Engineering (ICSE)*. IEEE, 1039–1049.

[93] Florian Klück, Martin Zimmermann, Franz Wotawa, and Mihai Nica. 2019. Genetic algorithm-based test parameter optimization for ADAS system testing. In *2019 IEEE 19th International Conference on Software Quality, Reliability and Security (QRS)*. IEEE, 418–425.

[94] A. Knauss, J. Schroder, C. Berger, and H. Eriksson. 2017. Software-Related Challenges of Testing Automated Vehicles. In *2017 IEEE/ACM 39th International Conference on Software Engineering Companion (ICSE-C)*. 328–330.

[95] Nathan Koenig and Andrew Howard. 2004. Design and use paradigms for gazebo, an open-source multi-robot simulator. In *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, Vol. 3. IEEE, 2149–2154.

[96] Twan Koolen and Robin Deits. 2019. Julia for robotics: Simulation and real-time control in a high-level programming language. In *2019 International Conference on Robotics and Automation (ICRA)*. IEEE, 604–611.

[97] Philip Koopman and Michael Wagner. 2016. Challenges in Autonomous Vehicle Testing and Validation. *SAE International Journal of Transportation Safety* 4 (04 2016), 15–24. https://doi.org/10.4271/2016-01-0128

[98] Panagiotis Kouvaros and Alessio Lomuscio. 2015. A counter abstraction technique for the verification of robot swarms. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 29.

[99] Panagiotis Kouvaros and Alessio Lomuscio. 2015. Verifying emergent properties of swarms. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[100] Panagiotis Kouvaros and Alessio Lomuscio. 2016. Formal verification of opinion formation in swarms. In *Proceedings of the 2016 International Conference on Autonomous Agents & Multiagent Systems*. 1200–1208.

[101] Panagiotis Kouvaros, Alessio Lomuscio, Edoardo Pirovano, and Hashan Punchihewa. 2019. Formal verification of open multi-agent systems. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems*. 179–187.

[102] Marta Kwiatkowska, Gethin Norman, and David Parker. 2011. PRISM 4.0: Verification of probabilistic real-time systems. In *International conference on computer aided verification*. Springer, 585–591.

[103] Kim G Larsen, Paul Pettersson, and Wang Yi. 1997. UPPAAL in a nutshell. *International journal on software tools for technology transfer* 1, 1-2 (1997), 134–152.

[104] Adrien Lasbouygues, Benoit Ropars, Robin Passama, David Andreu, and Lionel Lapierre. 2015. Atoms based control of mobile robots with Hardware-In-the-Loop validation. In *2015 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 1083–1090.

[105] Jannik Laval, Luc Fabresse, and Noury Bouraqadi. 2013. A methodology for testing mobile autonomous robots. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 1842–1847.

[106] Philippe Ledent, Anshul Paigwar, Alessandro Renzaglia, Radu Mateescu, and Christian Laugier. 2019. Formal Validation of Probabilistic Collision Risk Estimation for Autonomous Driving. In *2019 IEEE International Conference on Cybernetics and Intelligent Systems (CIS) and IEEE Conference on Robotics, Automation and Mechatronics (RAM)*. IEEE, 433–438.

[107] Li Li, Wu-Ling Huang, Yuehu Liu, Nan-Ning Zheng, and Fei-Yue Wang. 2016. Intelligence testing for autonomous vehicles: A new approach. *IEEE Transactions on Intelligent Vehicles* 1, 2 (2016), 158–166.

[108] Nan Li, Dave Oyler, Mengxuan Zhang, Yildiray Yildiz, Anouck Girard, and Ilya Kolmanovsky. 2016. Hierarchical reasoning game theory based approach for evaluation and testing of autonomous vehicle control systems. In *2016 IEEE 55th Conference on Decision and Control (CDC)*. IEEE, 727–733.

[109] Nan Li, Dave W Oyler, Mengxuan Zhang, Yildiray Yildiz, Ilya Kolmanovsky, and Anouck R Girard. 2017. Game theoretic modeling of driver and vehicle interactions for verification and validation of autonomous vehicle control systems. *IEEE Transactions on control systems technology* 26, 5 (2017), 1782–1797.

[110] Raimar Lill and Francesca Saglietti. 2014. Testing the cooperation of autonomous robotic agents. In *2014 9th International Conference on Software Engineering and Applications (ICSOFT-EA)*. IEEE, 287–296.

[111] Mikael Lindvall, Adam Porter, Gudjon Magnusson, and Christoph Schulze. 2017. Metamorphic model-based testing of autonomous systems. In *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*. IEEE, 35–41.

[112] Alessio Lomuscio and Jakub Michaliszyn. 2015. Verifying Multi-Agent Systems by Model Checking Three-valued Abstractions.. In *AAMAS*, Vol. 15. 189–198.

[113] Alessio Lomuscio and Edoardo Pirovano. 2019. A Counter Abstraction Technique for the Verification of Probabilistic Swarm Systems.. In *AAMAS*. 161–169.

[114] Alessio Lomuscio, Hongyang Qu, and Franco Raimondi. 2017. MCMAS: an open-source model checker for the verification of multi-agent systems. *International Journal on Software Tools for Technology Transfer* 19, 1 (2017), 9–30.

[115] Yu Lu, Hanlin Niu, Al Savvaris, and Antonios Tsourdos. 2016. Verifying collision avoidance behaviours for unmanned surface vehicles using probabilistic model checking. *IFAC-PapersOnLine* 49, 23 (2016), 127–132.

[116] Matt Luckcuck, Marie Farrell, Louise A. Dennis, Clare Dixon, and Michael Fisher. 2019. Formal Specification and Verification of Autonomous Robotic Systems: A Survey. *ACM Comput. Surv.* 52, 5 (2019), 100:1–100:41. https://doi.org/10.1145/3342355

[117] Israel Lugo-Cárdenas, Gerardo Flores, and Rogelio Lozano. 2014. The MAV3DSim: A simulation platform for research, education and validation of UAV controllers. *IFAC Proceedings Volumes* 47, 3 (2014), 713–717.

[118] Chenxia Luo, Rui Wang, Yu Jiang, Kang Yang, Yong Guan, Xiaojuan Li, and Zhiping Shi. 2018. Runtime verification of robots collision avoidance case study. In *2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC)*, Vol. 1. IEEE, 204–212.

[119] Damian M Lyons, Ronald C Arkin, Shu Jiang, Dagan Harrington, Feng Tang, and Peng Tang. 2015. Probabilistic verification of multi-robot missions in uncertain environments. In *2015 IEEE 27th International Conference on Tools with Artificial Intelligence (ICTAI)*. IEEE, 56–63.

[120] Damian M Lyons, Ronald C Arkin, Shu Jiang, Matt O'Brien, Feng Tang, and Peng Tang. 2017. Performance verification for robot missions in uncertain environments. *Robotics and Autonomous Systems* 98 (2017), 89–104.

[121] István Majzik, Oszkár Semeráth, Csaba Hajdu, Kristóf Marussy, Zoltán Szatmári, Zoltán Micskei, András Vörös, Aren A Babikian, and Dániel Varró. 2019. Towards system-level testing with coverage guarantees for autonomous vehicles. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems (MODELS)*. IEEE, 89–94.

[122] Michel Mamrot, Stefan Marchlewitz, Jan-Peter Nicklas, Petra Winzer, Thomas Tetzlaff, Philipp Kemper, and Ulf Witkowski. 2015. Model-based Test and Validation Support for Autonomous Mechatronic Systems. In *2015 IEEE International Conference on Systems, Man, and Cybernetics*. IEEE, 701–706.

[123] Md Abdullah Al Mamun, Christian Berger, and Jorgen Hansson. 2013. MDE-based sensor management and verification for a self-driving miniature vehicle. In *Proceedings of the 2013 ACM workshop on Domain-specific modeling*. 1–6.

[124] Musa Morena Marcusso Manhães, Sebastian A Scherer, Martin Voss, Luiz Ricardo Douat, and Thomas Rauschenbach. 2016. UUV simulator: A gazebo-based package for underwater intervention and multi-robot simulation. In *OCEANS 2016 MTS/IEEE Monterey*. IEEE, 1–8.

[125] Niloofar Mansoor, Jonathan A Saddler, Bruno Silva, Hamid Bagheri, Myra B Cohen, and Shane Farritor. 2018. Modeling and testing a family of surgical robots: an experience report. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 785–790.

[126] Casper Sloth Mariager, Daniel Kjaer Bonde Fischer, Jakob Kristiansen, and Matthias Rehm. 2019. Co-Designing and Field-Testing Adaptable Robots for Triggering Positive Social Interactions for Adolescents with Cerebral Palsy. In *2019 28th IEEE International Conference on Robot and Human Interactive Communication (RO-MAN)*. IEEE, 1–6.

[127] MATLAB. 2010. *version 7.10.0 (R2010a)*. The MathWorks Inc., Natick, Massachusetts.

[128] Steve McGuire, P Michael Furlong, Terry Fong, Christoffer Heckman, Daniel Szafir, Simon J Julier, and Nisar Ahmed. 2019. Everybody Needs Somebody Sometimes: Validation of Adaptive Recovery in Robotic Space Operations. *IEEE Robotics and Automation Letters* 4, 2 (2019), 1216–1223.

[129] Christopher Medrano-Berumen and Mustafaff Ilhan Akbaş. 2019. Abstract simulation scenario generation for autonomous vehicle verification. In *2019 SoutheastCon*. IEEE, 1–6.

[130] Alvaro Miyazawa, Pedro Ribeiro, Wei Li, ALC Cavalcanti, Jon Timmis, and JCP Woodcock. 2016. RoboChart: a state-machine notation for modelling and verification of mobile and autonomous robots. *Tech. Rep.* (2016).

[131] Maurizio Mongelli, Marco Muselli, Andrea Scorzoni, and Enrico Ferrari. 2019. Accellerating prism validation of vehicle platooning through machine learning. In *2019 4th International Conference on System Reliability and Safety (ICSRS)*. IEEE, 452–456.

[132] Shahabuddin Muhammad, Nazeeruddin Mohammad, Abul Bashar, and Majid Ali Khan. 2019. Designing human assisted wireless sensor and robot networks using probabilistic model checking. *Journal of Intelligent & Robotic Systems* 94, 3-4 (2019), 687–709.

[133] Galen E Mullins, Paul G Stankiewicz, R Chad Hawthorne, and Satyandra K Gupta. 2018. Adaptive generation of challenging scenarios for testing and evaluation of autonomous vehicles. *Journal of Systems and Software* 137 (2018), 197–215.

[134] Adnan Munawar and Gregory S Fischer. 2019. An asynchronous multi-body simulation framework for real-time dynamics, haptics and learning with application to surgical robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*.

[135] Florian Mutter, Stefanie Gareis, Bernhard Schätz, Andreas Bayha, Franziska Grüneis, Michael Kanis, and Dagmar Koss. 2011. Model-driven in-the-loop validation: Simulation-based testing of UAV software using virtual environments. In *2011 18th IEEE International Conference and Workshops on Engineering of Computer-Based Systems*. IEEE, 269–275.

[136] Frederik Naujoks, Sebastian Hergeth, Katharina Wiedemann, Nadja Schömig, and Andreas Keinath. 2018. Use cases for assessing, testing, and validating the human machine interface of automated driving systems. In *Proceedings of the Human Factors and Ergonomics Society Annual Meeting*, Vol. 62. SAGE Publications Sage CA: Los Angeles, CA, 1873–1877.

[137] Cu D Nguyen, Simon Miles, Anna Perini, Paolo Tonella, Mark Harman, and Michael Luck. 2012. Evolutionary testing of autonomous software agents. *Autonomous Agents and Multi-Agent Systems* 25, 2 (2012), 260–283.

[138] Matthew O'Kelly, Houssam Abbas, Sicun Gao, Shin'ichi Shiraishi, Shinpei Kato, and Rahul Mangharam. 2016. APEX: Autonomous vehicle plan verification and execution. (2016).

[139] Stephan Opfer, Stefan Niemczyk, and Kurt Geihs. 2016. Multi-agent plan verification with answer set programming. In *Proceedings of the 3rd Workshop on Model-Driven Robot Software Engineering*. 32–39.

[140] Matthew O'Brien, Ronald C Arkin, Dagan Harrington, Damian Lyons, and Shu Jiang. 2014. Automatic verification of autonomous robot missions. In *International Conference on Simulation, Modeling, and Programming for Autonomous Robots*. Springer, 462–473.

[141] Jisun Park, Mingyun Wen, Yunsick Sung, and Kyungeun Cho. 2019. Multiple event-based simulation scenario generation approach for autonomous vehicle smart sensors and devices. *Sensors* 19, 20 (2019), 4456.

[142] Corina S Pasareanu, Divya Gopinath, and Huafeng Yu. 2018. Compositional Verification for Autonomous Systems with Deep Learning Components. *arXiv preprint arXiv:1810.08303* (2018).

[143] Shashank Pathak, Giorgio Metta, and Armando Tacchella. 2014. Is verification a requisite for safe adaptive robots?. In *2014 IEEE International Conference on Systems, Man, and Cybernetics (SMC)*. IEEE, 3399–3402.

[144] José LF Pereira and Rosaldo JF Rossetti. 2012. An integrated architecture for autonomous vehicles simulation. In *Proceedings of the 27th annual ACM symposium on applied computing*. 286–292.

[145] Mauro Pezze and Michal Young. 2007. . Wiley.

[146] Javier Poncela and MC Aguayo-Torres. 2013. A Framework for Testing of Wireless Underwater Robots. *Wireless personal communications* 70, 3 (2013), 1171–1181.

[147] David Porfirio, Allison Sauppé, Aws Albarghouthi, and Bilge Mutlu. 2018. Authoring and verifying human-robot interactions. In *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. 75–86.

[148] Martin Proetzsch, Fabian Zimmermann, Robert Eschbach, Johannes Kloos, and Karsten Berns. 2010. A systematic testing approach for autonomous mobile robots using domain-specific languages. In *Annual Conference on Artificial Intelligence*. Springer, 317–324.

[149] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. 2009. ROS: an open-source Robot Operating System. In *ICRA workshop on open source software*, Vol. 3. Kobe, Japan, 5.

[150] Nijat Rajabli, Francesco Flammini, Roberto Nardone, and Valeria Vittorini. 2021. Software Verification and Validation of Safe Autonomous Cars: A Systematic Literature Review. *IEEE Access* 9 (2021), 4797–4819. https://doi.org/10.1109/ACCESS.2020.3048047

[151] Arvind Ramanathan, Laura L Pullum, Faraz Hussain, Dwaipayan Chakrabarty, and Sumit Kumar Jha. 2016. Integrating symbolic and statistical methods for testing intelligent systems: Applications to machine learning and computer vision. In *2016 Design, Automation & Test in Europe Conference & Exhibition (DATE)*. IEEE, 786–791.

[152] Q. Rao and J. Frtunikj. 2018. Deep Learning for Self-Driving Cars: Chances and Challenges. In *2018 IEEE/ACM 1st International Workshop on Software Engineering for AI in Autonomous Systems (SEFAIAS)*. IEE.

[153] Signe A. Redfield and Mae L. Seto. 2017. Verification Challenges for Autonomous Systems. In *Autonomy and Artificial Intelligence: A Threat or Savior?*, William F. Lawless, Ranjeev Mittu, Donald Sofge, and Stephen Russell (Eds.). Springer International Publishing, 103–127. https://doi.org/10.1007/978-3-319-59719-5_5

[154] Pedro Ribeiro, Alvaro Miyazawa, Wei Li, Ana Cavalcanti, and Jon Timmis. 2017. Modelling and verification of timed robotic controllers. In *International Conference on Integrated Formal Methods*. Springer, 18–33.

[155] Sergio Rico, Emelie Engström, and Martin Höst. 2019. A Taxonomy for Improving Industry-Academia Communication in IoT Vulnerability Management. In *Proceedings of the 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 38–45.

[156] Eric Rohmer, Surya PN Singh, and Marc Freese. 2013. V-REP: A versatile and scalable robot simulation framework. In *2013 IEEE/RSJ International Conference on Intelligent Robots and Systems*. IEEE, 1321–1326.

[157] Martijn Rooker, Pablo Horstrand, Aythami Salvador Rodriguez, Sebastian Lopez, Roberto Sarmiento, Jose Lopez, Ray Alejandro Lattarulo, Joshue Manuel Perez Rastelli, Zora Slavik, David Pereira, et al. 2018. Towards improved validation of autonomous systems for smart farming. In *Smart Farming Workshop*.

[158] Gregg Rothermel, Roland H. Untch, Chengyun Chu, and Mary Jean Harrold. [n. d.]. Test Case Prioritization: An Empirical Study. In *Proc. of the 1999 International Conference on Software Maintenance, ICSM 1999*.

[159] Sasha Rubin. 2015. Parameterised verification of autonomous mobile-agents in static but unknown environments. In *Proceedings of the 2015 International Conference on Autonomous Agents and Multiagent Systems*. 199–208.

[160] Peter AM Ruijten, Antal Haans, Jaap Ham, and Cees JH Midden. 2019. Perceived human-likeness of social robots: testing the Rasch model as a method for measuring anthropomorphism. *International Journal of Social Robotics* 11, 3 (2019), 477–494.

[161] Rim Saddem, Olivier Naud, Karen Godary Dejean, and Didier Crestani. 2017. Decomposing the model-checking of mobile robotics actions on a grid. *IFAC-PapersOnLine* 50, 1 (2017), 11156–11162.

[162] Francesca Saglietti and Matthias Meitner. 2016. Model-driven structural and statistical testing of robot cooperation and reconfiguration. In *Proceedings of the 3rd Workshop on Model-Driven Robot Software Engineering*. 17–23.

[163] Francesca Saglietti, Stefan Winzinger, and Raimar Lill. 2014. Reconfiguration testing for cooperating autonomous agents. In *International Conference on Computer Safety, Reliability, and Security*. Springer, 144–155.

[164] André Santos, Alcino Cunha, and Nuno Macedo. 2018. Property-based testing for the robot operating system. In *Proceedings of the 9th ACM SIGSOFT International Workshop on Automating TEST Case Design, Selection, and Evaluation*. 56–62.

[165] Ichiro Satoh. 2018. An approach for testing software on networked transport robots. In *2018 14th IEEE International Workshop on Factory Communication Systems (WFCS)*. IEEE, 1–4.

[166] Ichiro Satoh. 2019. Developing and Testing Networked Software for Moving Robots.. In *ENASE*. 315–321.

[167] Hans-Peter Schöner. 2018. Simulation in development and testing of autonomous vehicles. In *18. Internationales Stuttgarter Symposium*. Springer, 1083–1095.

[168] Yuvaraj Selvaraj, Wolfgang Ahrendt, and Martin Fabian. 2019. Verification of Decision Making Software in an Autonomous Vehicle: An Industrial Case Study. In *International Workshop on Formal Methods for Industrial Critical Systems*. Springer, 143–159.

[169] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. 2018. Airsim: High-fidelity visual and physical simulation for autonomous vehicles. In *Field and service robotics*. Springer, 621–635.

[170] Weijing Shi, Mohamed Baker Alawieh, Xin Li, Huafeng Yu, Nikos Arechiga, and Nobuyuki Tomatsu. 2016. Efficient statistical validation of machine learning systems for autonomous driving. In *Proceedings of the 35th International Conference on Computer-Aided Design*. 1–8.

[171] Christoph Sippl, Florian Bock, David Wittmann, Harald Altinger, and Reinhard German. 2016. From simulation data to test cases for fully automated driving and ADAS. In *IFIP International Conference on Testing Software and Systems*. Springer, 191–206.

[172] Gopinadh Sirigineedi, Antonios Tsourdos, Brian A White, and Rafał Żbikowski. 2011. Kripke modelling and verification of temporal specifications of a multiple UAV system. *Annals of Mathematics and Artificial Intelligence* 63, 1 (2011), 31–52.

[173] Michał Siwek, Leszek Baranowski, Jarosław Panasiuk, and Wojciech Kaczmarek. 2019. Modeling and simulation of movement of dispersed group of mobile robots using Simscape multibody software. In *AIP Conference Proceedings*, Vol. 2078. AIP Publishing LLC, 020045.

[174] Marc Spislaender and Francesca Saglietti. 2018. Evidence-Based Verification of Safety Properties Concerning the Cooperation of Autonomous Agents. In *2018 44th Euromicro Conference on Software Engineering and Advanced Applications (SEAA)*. IEEE, 81–88.

[175] Tomoo Sumida, Hiroyuki Suzuki, Sho Sei Shun, Kazuhito Omaki, Takaaki Goto, and Kensei Tsuchida. 2017. FDR verification of a system involving a robot climbing stairs. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science (ICIS)*. IEEE, 875–878.

[176] Xiaowu Sun, Haitham Khedr, and Yasser Shoukry. 2019. Formal verification of neural network controlled autonomous systems. In *Proceedings of the 22nd ACM International Conference on Hybrid Systems: Computation and Control*. 147–156.

[177] Zsolt Szalay, Mátyás Szalai, Bálint Tóth, Tamás Tettamanti, and Viktor Tihanyi. 2019. Proof of concept for Scenario-in-the-Loop (SciL) testing for autonomous vehicle technology. In *2019 IEEE International Conference on Connected Vehicles and Expo (ICCVE)*. IEEE, 1–5.

[178] Jianbo Tao, Yihao Li, Franz Wotawa, Hermann Felbinger, and Mihai Nica. 2019. On the industrial application of combinatorial testing for autonomous driving functions. In *2019 IEEE International Conference on Software Testing, Verification and Validation Workshops (ICSTW)*. IEEE, 234–240.

[179] Mugur Tatar. 2015. Enhancing ADAS test and validation with automated search for critical situations. In *Driving Simulation Conference (DSC)*.

[180] Yuchi Tian, Kexin Pei, Suman Jana, and Baishakhi Ray. 2018. Deeptest: Automated testing of deep-neural-network-driven autonomous cars. In *Proceedings of the 40th international conference on software engineering*. 303–314.

[181] Thomas Tosik, Jasper Schwinghammer, Mandy Jane Feldvoß, John Paul Jonte, Arne Brech, and Erik Maehle. 2016. MARS: A simulation environment for marine swarm robotics and environmental monitoring. In *OCEANS 2016-Shanghai*. IEEE, 1–6.

[182] Tarik Tosun, Gangyuan Jing, Hadas Kress-Gazit, and Mark Yim. 2018. Computer-aided compositional design and verification for modular robots. In *Robotics Research*. Springer, 237–252.

[183] Garazi Juez Uriagereka, Estibaliz Amparan, Cristina Martinez Martinez, Jabier Martinez, Aurelien Ibanez, Matteo Morelli, Ansgar Radermacher, and Huascar Espinoza. 2019. Design-Time Safety Assessment of Robotic Systems Using Fault Injection Simulation in a Model-Driven Approach. In *2019 ACM/IEEE 22nd International Conference on Model Driven Engineering Languages and Systems Companion (MODELS-C)*. IEEE, 577–586.

[184] Vandi Verma and Chris Leger. 2019. SSim: NASA Mars Rover Robotics Flight Software Simulation. In *2019 IEEE Aerospace Conference*. IEEE, 1–11.

[185] Federico Vicentini, Mehrnoosh Askarpour, Matteo G Rossi, and Dino Mandrioli. 2019. Safety assessment of collaborative robotics through automated formal verification. *IEEE Transactions on Robotics* 36, 1 (2019), 42–61.

[186] Harsha Jakkanahalli Vishnukumar, Björn Butting, Christian Müller, and Eric Sax. 2017. Machine learning and deep neural network—Artificial intelligence core for lab and real-world test and validation for ADAS and autonomous vehicles: AI for efficient and quality test and validation. In *2017 Intelligent Systems Conference (IntelliSys)*. IEEE, 714–721.

[187] Dennis Walter, Holger Täubig, and Christoph Lüth. 2010. Experiences in applying formal verification in robotics. In *International Conference on Computer Safety, Reliability, and Security*. Springer, 347–360.

[188] Kai Wang and JC Cheng. 2019. Integrating Hardware-In-the-Loop Simulation and BIM for Planning UAV-Based As-Built MEP Inspection with Deep Learning Techniques. In *Proceedings of the 36th International Symposium on Automation and Robotics in Construction*. 310–316.

[189] Rui Wang, Yingxia Wei, Houbing Song, Yu Jiang, Yong Guan, Xiaoyu Song, and Xiaojuan Li. 2018. From offline towards real-time verification for robot systems. *IEEE Transactions on Industrial Informatics* 14, 4 (2018), 1712–1721.

[190] Kosuke Watanabe, Eunsuk Kang, Chung-Wei Lin, and Shinichi Shiraishi. 2018. Runtime monitoring for safety of intelligent vehicles. In *Proceedings of the 55th Annual Design Automation Conference*. 1–6.

[191] Matt Webster, Clare Dixon, Michael Fisher, Maha Salem, Joe Saunders, Kheng Lee Koay, Kerstin Dautenhahn, and Joan Saez-Pons. 2015. Toward reliable autonomous robotic assistants through formal verification: A case study. *IEEE Transactions on Human-Machine Systems* 46, 2 (2015), 186–196.

[192] Matt Webster, Maha Salem, Clare Dixon, Michael Fisher, and Kerstin Dautenhahn. 2014. Formal verification of an autonomous personal robotic assistant. (2014).

[193] Dennis Leroy Wigand, Pouya Mohammadi, Enrico Mingo Hoffman, Nikos G Tsagarakis, Jochen J Steil, and Sebastian Wrede. 2018. An open-source architecture for simulation, execution and analysis of real-time robotics systems. In *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. IEEE, 93–100.

[194] Tichakorn Wongpiromsarn, Sayan Mitra, Andrew Lamperski, and Richard M Murray. 2012. Verification of periodically controlled hybrid systems: Application to an autonomous vehicle. *ACM Transactions on Embedded Computing Systems (TECS)* 11, S2 (2012), 1–24.

[195] Bingqing Xu, Qin Li, Tong Guo, Yi Ao, and Dehui Du. 2019. A quantitative safety verification approach for the decision-making process of autonomous driving. In *2019 International Symposium on Theoretical Aspects of Software Engineering (TASE)*. IEEE, 128–135.

[196] Bingqing Xu, Qin Li, Tong Guo, and Dehui Du. 2019. A Scenario-Based Approach for Formal Modelling and Verification of Safety Properties in Automated Driving. *IEEE Access* 7 (2019), 140566–140587.

[197] Wing Lok Yeung. 2011. Behavioral modeling and verification of multi-agent systems for manufacturing control. *Expert Systems with applications* 38, 11 (2011), 13555–13562.

[198] Levent Yilmaz. 2017. Verification and validation of ethical decision-making in autonomous systems. In *Proceedings of the Symposium on Modeling and Simulation of Complexity in Intelligent, Adaptive and Autonomous Systems*. 1–12.

[199] Fu Yujian and Drabo Mebougna. 2014. formal Modeling and verification of dynamic reconfiguration of autonomous robotics systems. In *Proceedings of the International Conference on Embedded Systems and Applications (ESA)*, Vol. 2. 14.

[200] Sunkil Yun, Takaaki Teshima, and Hidekazu Nishimura. 2019. Human–Machine Interface Design and Verification for an Automated Driving System Using System Model and Driving Simulator. *IEEE Consumer Electronics Magazine* 8, 5 (2019), 92–98.

[201] Chi Zhang, Yuehu Liu, Danchen Zhao, and Yuanqi Su. 2014. RoadView: A traffic scene simulator for autonomous vehicle simulation testing. In *17th International IEEE Conference on Intelligent Transportation Systems (ITSC)*. IEEE, 1160–1165.

[202] Xiaoyang Zhang, Hongpeng Wang, Jingtai Liu, and Haifeng Li. 2019. CyberEarth: a Virtual Simulation Platform for Robotics and Cyber-Physical Systems. In *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*. IEEE, 858–863.

[203] Xingyu Zhao, Matt Osborne, Jenny Lantair, Valentin Robu, David Flynn, Xiaowei Huang, Michael Fisher, Fabio Papacchini, and Angelo Ferrando. 2019. Towards Integrating Formal Verification of Autonomous Robots with Battery Prognostics and Health Management. In *International Conference on Software Engineering and Formal Methods*. Springer, 105–124.

[204] Xingyu Zhao, Valentin Robu, David Flynn, Fateme Dinmohammadi, Michael Fisher, and Matt Webster. 2019. Probabilistic model checking of robots deployed in extreme environments. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 33. 8066–8074.

[205] Xingyu Zhao, Valentin Robu, David Flynn, Kizito Salako, and Lorenzo Strigini. 2019. Assessing the safety and reliability of autonomous vehicles from road testing. In *2019 IEEE 30th International Symposium on Software Reliability Engineering (ISSRE)*. IEEE, 13–23.

[206] Jinwei Zhou, Roman Schmied, Alexander Sandalek, Helmut Kokal, and Luigi del Re. 2016. A framework for virtual testing of adas. *SAE International Journal of Passenger Cars-Electronic and Electrical Systems* 9, 2016-01-0049 (2016), 66–73.

[207] Marc René Zofka, Marc Essinger, Tobias Fleck, Ralf Kohlhaas, and J Marius Zöllner. 2018. The sleepwalker framework: Verification and validation of autonomous vehicles by mixed reality lidar stimulation. In *2018 IEEE International Conference on Simulation, Modeling, and Programming for Autonomous Robots (SIMPAR)*. IEEE, 151–157.