Simulation-based testing to improve safety of autonomous robots

Luca Vittorio Sartori
LAAS-CNRS, University of Toulouse
Toulouse, France
lsartori@laas.fr

***Abstract*—Autonomous systems are becoming increasingly popular among industries as well as end-users, and are deployed in numerous tasks. To improve their reliability and to avoid critical failures that impact safety, their testing aims at ensuring that their behavior and decisions are acceptable even in scenarios that have not been foreseen by the developers. As testing of these systems is usually done through field testing, which is costly and is limited in the reproducible scenarios, system-level pre-validation can be done in virtual worlds through simulation, to discover faults and fix them before deploying the system in the real world. However, there is no current standard procedure to conduct simulation-based testing and to ensure satisfying coverage of the most critical scenarios for the system under test (SUT). The aim of this experimental work is to improve and automate the steps of the simulation-based testing related to the generation and selection of test inputs, the exploitation of the results, and the incorporation of dynamic agents in the tests.**

***Index Terms*—Simulation, testing, test oracle, industrial case study, safety, autonomous systems, world generation**

I. Introduction

Autonomous systems, such as autonomous vehicles and robots, take decisions and act without human intervention, and are being developed by the industry for a wide spectrum of applications, from transportation purposes such as delivery robots, to agricultural drones that monitor crop growth. The failure of autonomous systems can have a serious impact on their mission, performance or safety. Thus, they should be thoroughly tested to prevent failures, covering diverse and critical situations. As of now, most systems are tested through field testing, a method that is costly, time-consuming, limited in the reproducible scenarios, and risky in case of non-acceptable behaviors. To mitigate these issues, their software can be pre-validated using simulation-based testing, for example, with model-in-the-loop (MiL) or software-in-the-loop (SiL) approaches, simulating the behavior of the system inside a virtual environment, which resembles the real environment where the system will operate. An example of a general testing loop for SiL is depicted in Fig. 1. In this context, a test case becomes the combination of assigning a mission to the system and generating a virtual world, where the mission will take place. Simulation-based testing cannot substitute field testing, as it is limited in how it simulates physics and it can be computationally expensive. However, the two testing approaches complement each other and can guarantee better testing of the system software. Since just a portion of test cases will lead to failures, an important issue is the selection of test cases to run in the simulation, as the search space, i.e., the set of possible test cases generated by the generation parameters, e.g., size of the world, number of obstacles, increases exponentially in relation to the number of parameters that model the virtual worlds. Consequently, executing all the test cases is unfeasible and not efficient. Moreover, revealing as many failures as possible in simulation is key to discover the causing faults, which can then be analyzed and fixed before the robot is field tested. On this subject, at LAAS-CNRS, there is a framework of ongoing work [1]–[3]. The work has focused on the generation of virtual worlds, has demonstrated that it is possible to find bugs in low-fidelity simulation, and has shown that it is possible to coarsely adjust the difficulty level of the worlds for the system under test (SUT). This thesis started in March 2019, is inserted in the above-mentioned framework, and aims at answering the following Research Questions (RQs):

- RQ1: How to guide the selection of test cases in order to maximize the discovery of faults, taking into account test objectives and the analysis of previous runs.
- RQ2: How to exploit the results to characterize the high-level properties of the scenarios exhibiting failures.
- RQ3: How to incorporate a model of dynamic agents.

The thesis is experimental, includes two industrial case studies with mobile robots, and involves worlds with dynamic agents.

After outlining in Section II the body of literature relevant to this work, the current challenges are presented in Section III. Section IV describes the proposed methods to face the challenges and Section V concludes the paper and details the work plan.

II. Related work

The testing community is researching how to model and generate content, what to test, with which methods to test, how to evaluate the outputs of the tests, and how to automate the procedures. The selection of the simulator has also to be taken into account, as different simulators have different capabilities. Moreover, it is possible to customize the physical fidelity of the simulator, with a higher fidelity requiring more computational time and a bigger effort in the development of the simulator. Since the search budget, i.e., the time allotted for testing, is a constraint imposed by the company developing the system, simulations can have low physical fidelity, prioritizing

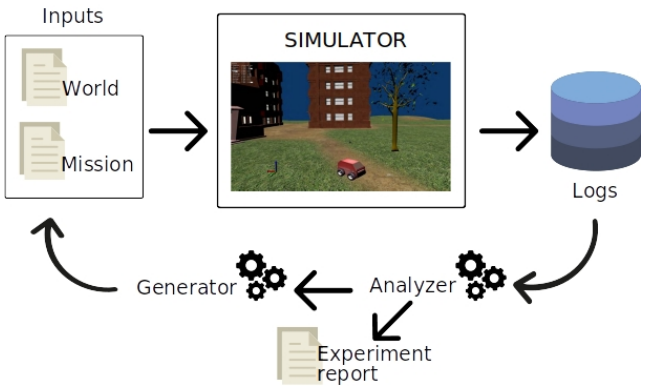

Fig. 1. Conceptual view of the testing loop, from [1].

the number of cases executed per search budget. There are various robotics simulators such as GAZEBO [4] and MORSE [5], and game engines like Unity [6] and Unreal Engine [7] that have been used to conduct simulations: a comparison of which is given in [8]. In [9], Procedural Content Generation (PCG)—an automated technique borrowed from the video games field [10]—is used to generate 2D environments to test the path-following algorithm with collision avoidance of a mobile robot, demonstrating that bugs can be found even with low physical fidelity. [11] came to the same conclusion, showing that the majority of the bugs of the ArduPilot System are exposed by trivial conditions, and do not require the reproduction of complex physical phenomena to be revealed. Previous work at LAAS reinforces this idea, as in the study on the navigation software of the mobile robot Mana [2], where out of 33 bugs, the only bug that needed high fidelity to be replicated was related to mechanical vibration, which was not included in the simulation. The same authors in [1] investigate the system level testing of Mana. In that experiment, the generation of test cases uses a world model specified in a UML class diagram, which manages the high-level generation parameters—the size, percentage of obstruction, and smoothness—demonstrating that it is possible to coarsely control the difficulty level of the test cases.

Parametrized world models define a search space for the selection of test cases, suggesting the use of search-based software testing (SBST). SBST, a subset of search-based software engineering [12], uses meta-heuristic to find satisfying solutions to a test objective. The quality of the solution, i.e., the generated test cases, is assessed through a fitness function. SBST has been applied to the generation of test cases for autonomous systems, with the heuristics ranging from a simple random search [9] to genetic algorithms (GA) [13], or to a hybridization of GA and machine learning (ML) [14]. In the latter work, the hybridization aims to characterize subsets—critical regions—of the search space with a higher probability of containing critical test scenarios. Specifically, the proposed algorithm consists of two nested loops, where the results of GA from the inner loop allow ML in the outer loop to identify combinations of parameters that are more likely to induce critical scenarios, yielding critical regions that deserve more exploration by GA, which refines the learned critical regions, and so on.

After the simulations are completed, assigning a pass/fail verdict to a test case is specifically challenging for autonomous systems, because the decisional aspects make it difficult to determine the right output for a given input, which is known as the test oracle problem. Moreover, the test oracle is usually based on a set of high-level requirements, like the absence of collision. Previous work at LAAS on the Mana robot identified five broad classes of requirements to check: (i) requirements attached to mission phases, (ii) thresholds related to robot movement, (iii) absence of catastrophic events, (iv) requirements attached to error reports, v) perception requirements. Other authors have considered metamorphic properties relating a test case to follow-up cases derived from it. In [15], metamorphic testing is applied to autonomous drones, by rotating a generated world to create follow-up cases. The drone should behave the same way in the original and rotated worlds. The comparison involves the path that the drone takes, the mission success, and a sequence of system states reconstructed from sensors values, giving a high-level view of the evolution of the system. Finally, some approaches consider a quantitative assessment of test results rather than a binary pass/fail verdict. For the cleaning agent in [13], quality thresholds are proposed, which are for example associated with battery levels. In [16], for cyber-physical system models, a quantitative test oracle provides a measure of the degree of satisfaction or violation of requirements.

III. Challenges

In this section, the RQs and the challenges that they address are framed in the context of the testing steps.

A. Search-based generation of test cases (RQ1)

RQ1 is located after the definition of the search space, and focuses on finding an efficient search-based technique to explore the search space, given the budget constraints. RQ1 faces the following challenges: i) How to exploit knowledge from previous test runs to guide the search, as it is preferable to avoid running test cases that have a low probability of failing, accelerating the convergence to cases that expose failures. ii) How to accommodate multiple and conflicting objectives, because as their number increases, it is not possible to find a single solution that satisfies all of them, hence the need for a trade-off. The issue becomes to find a set of test cases where it is not possible to maximize more an objective without decreasing another one, i.e., the issue is to find the Pareto front. iii) How to accommodate non-determinism, that is, the property of obtaining different results for repeated runs of the same test case, as in [1], due to the decisional capabilities of the robot. Thus, multiple runs per test case are necessary to provide sufficient data to statistically measure the fitness [13]. Additionally, different missions can have different degrees of non-determinism, requiring a different number of runs. iv) How to select the algorithm among the many

existing ones. When selecting the algorithm to analyze the search space, there are two key concepts to take into account: exploration—the diversity of solutions—and exploitation—the convergence to the optima of the search space. According to the search budget, different families of algorithms can be used based on their ability to show more diverse failures or more severe failures. It may be possible with a restricted time budget that a simple search algorithm is more efficient than a more sophisticated meta-heuristic algorithm at finding the optima.

B. Exploitation of raw test results (RQ2)

After the algorithm has been selected, the test cases executed, and the results collected, RQ2 focuses on understanding and describing what occurred during the simulation, as it is crucial to communicate to the companies why and how the robot failed. Since the search-based process may produce many failing test cases that could actually be variants of the same failure scenario, the first challenge is how to recognize them and group them in a single set. The second challenge is finding a mean to extract some key information from the raw data of the runs with failures, to simplify the diagnosis of the results. For these reasons, RQ2 aims at extracting a high-level view, at providing a better description from the states of the system, at mitigating the effect of non-determinism, and at helping the engineers to focus on the subset of data that gives meaningful information about the failures.

C. Dynamic agents in world models (RQ3)

RQ3 is concerned with the generation step, so before RQ1 in the testing procedure, but it will be investigated chronologically later in this PhD thesis, hence, it is positioned as the third RQ. RQ3 stems from the need to augment static worlds with dynamic agents, which have to be modelled in the generation step. Moreover, the dynamic agents will influence the execution step and will complicate the definition of the test oracle. While the work at LAAS has focused on static obstacles, this thesis investigates the introduction of dynamic agents—such as mobile robots and humans—their missions, and events triggered by specific conditions. The challenge is how to model dynamic agents to maximize the information obtained on the SUT from their introduction in the simulation. They also must create coherent and realistic situations, or they will lower the probability of finding failures, since their introduction will enlarge the search space.

IV. Proposed methods

At the current stage of the thesis, for the RQ1, a promising direction to maximize the number of critical test cases to run lies in meta-heuristic, specifically in evolutionary algorithms such as GAs, as it is possible to tune them to balance exploration and exploitation. Using ML to guide them is also a promising solution.

With the final goal of better describing the scene and clarifying what is occurring in the simulation, RQ2 is addressed focusing on finding patterns in the scenarios that give a fail verdict, which is also connected to the selection problem. Identifying critical states and finding high-level parameters that describe the situations requires a posteriori analysis, inputs analysis, geometrical properties description, and reverse engineering of the scenarios. Hence, ML and state diagrams are interesting directions. In parallel, the current test oracle used at LAAS will be improved with a quantitative approach describing the degree of failure, as defined by the company, or how far the result is from a failing test case. This approach, which includes multiple properties to check, gives a global view and a better description of the circumstances during the simulation.

For RQ3 a model for dynamic agents will be created, then it will be upgraded with more articulated missions, and at last a probabilistic model of the behavior based on the mission will be added. Features will be modelled to maximize the interaction with the SUT, to stress it, and to solicit a state change of the SUT to a critical state.

This work will be heavily experimental. The proposed approaches will be consolidated and assessed by means of case studies. One academic and two industrial case studies are available for this work:

- OSMOSIS [17], a simple academic case of a robot for airport light inspection.
- An agricultural weeding robot named Oz, visible in Fig. 2 and produced by Naïo, a French company.
- Automated Guided Carts (AGCs), studied by Sick AG, a company based in Germany.

As in any empirical study, there are threats to validity. In construct validity the threats relate to the identification of the faults highlighted by the detected failures. Two secondments in companies, of three months each, have been planned to ensure proper communication and discussion between the researcher and the engineers responsible for the development and debugging. External validity relates to generalizing the results from the two mobile robots cases. The cases of the robots under test exemplify two complementary industrial cases, as they have opposite characteristics: one robot vs multiple robots, outdoor open space vs indoor closed space, and without agents vs with agents. In testing, the approaches for the generation, selection, and analysis that will be adopted are general and can be applied to other cases.

Fig. 2. The weeding robot Oz. Source: Naïo Technologies.

V. Conclusions and work plan

Autonomous systems will be increasingly complex and widespread in the next decades, thus, minimizing their non-acceptable behaviors and assuring safety though testing is fundamental to accelerate their adoption and acceptance into the society. Their testing poses challenges from the generation of the test inputs to the evaluation of the outputs. This research work follows a three-year schedule, focuses on mobile robots, and aims at providing sound approaches to automate software testing, reducing the costs and increasing the safety of the robots software.

The expected contribution of this thesis is a complete and generic framework to carry out simulation-based testing of autonomous robots in virtual worlds. The automated world generation process will be guided by test objectives and analysis of previous test runs. The work will help testing the robots software of the companies involved, according to the following schedule.

At the time of writing, the research work is still at the beginning, and its work plan follows a three-year schedule, starting from March 2019:

1st year

- Literature review and improvement of the world and mission generation model of OSMOSIS.
- Evaluation of meta-heuristic approaches to guide the generation and selection of test cases for OSMOSIS, and comparison with random search—the baseline.
- Secondment at Naïo to understand the behavioral requirements of the weeding robot.

2nd year

- Experimental comparison between the meta-heuristic approaches and field testing in Naïo's case, and investigation on the quantitative oracle.
- Work on the exploitation of the test results for clustering, through modeling of high-level properties, to characterize sets of failing scenarios. Comparison of the fault diagnosis performance of this approach vs the approach without clustering.
- Secondment at Sick AG regarding AGCs and development of a human behavior model for cases relevant to the company.

3rd year

- Introduction of dynamic agents in the generation model and evaluation of faults diagnosis performance vs the case of simulation with static obstacles.
- Experimental comparison between the simulation-based approach and field testing for Sick AG.
- Finalization of the testing framework and assessment of the trust and safety achieved with the proposed framework.

Acknowledgment

This project has received funding from the European Union's EU Framework Programme for Research and Innovation Horizon 2020 under Grant Agreement No. 812.788. The author thanks Hélène Waeselynck, Jérémie Guiochet, Magnus Albert, and Rob Alexander for their guidance during the first months of this PhD.

References

[1] T. Sotiropoulos, J. Guiochet, F. Ingrand, and H. Waeselynck, "Virtual Worlds for Testing Robot Navigation: A Study on the Difficulty Level," in *2016 12th European Dependable Computing Conference (EDCC)*, Sep. 2016, pp. 153–160.

[2] T. Sotiropoulos, H. Waeselynck, J. Guiochet, and F. Ingrand, "Can Robot Navigation Bugs Be Found in Simulation? An Exploratory Study," in *2017 IEEE International Conference on Software Quality, Reliability and Security (QRS)*, Jul. 2017, pp. 150–159.

[3] C. Robert, "First Insights into Testing Autonomous Robot in Virtual Worlds," in *2017 IEEE International Symposium on Software Reliability Engineering Workshops (ISSREW)*, Oct. 2017, pp. 112–115.

[4] N. Koenig and A. Howard, "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, Sep. 2004, pp. 2149–2154 vol.3.

[5] G. Echeverria, N. Lassabe, A. Degroote, and S. Lemaignan, "Modular open robots simulation engine: MORSE," in *2011 IEEE International Conference on Robotics and Automation*, May 2011, pp. 46–51.

[6] "Unity." [Online]. Available: https://unity.com/ (Accessed 2019-07-25).

[7] "Unreal Engine | What is Unreal Engine 4." [Online]. Available: https://www.unrealengine.com/ (Accessed 2019-07-25).

[8] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A Systematic Review of Perception System and Simulators for Autonomous Vehicles Research," *Sensors*, vol. 19, no. 3, p. 648, Jan. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/3/648

[9] J. Arnold and R. Alexander, "Testing Autonomous Robot Control Software Using Procedural Content Generation," in *Computer Safety, Reliability, and Security*, ser. Lecture Notes in Computer Science, F. Bitsch, J. Guiochet, and M. Kaâniche, Eds. Springer Berlin Heidelberg, 2013, pp. 33–44.

[10] J. Togelius, G. N. Yannakakis, K. O. Stanley, and C. Browne, "Search-Based Procedural Content Generation: A Taxonomy and Survey," *IEEE Transactions on Computational Intelligence and AI in Games*, vol. 3, no. 3, pp. 172–186, Sep. 2011.

[11] C. S. Timperley, A. Afzal, D. S. Katz, J. M. Hernandez, and C. L. Goues, "Crashing Simulated Planes is Cheap: Can Simulation Detect Robotics Bugs Early?" in *2018 IEEE 11th International Conference on Software Testing, Verification and Validation (ICST)*, Apr. 2018, pp. 331–342.

[12] M. Harman, S. A. Mansouri, and Y. Zhang, "Search-based Software Engineering: Trends, Techniques and Applications," *ACM Comput. Surv.*, vol. 45, no. 1, pp. 11:1–11:61, Dec. 2012. [Online]. Available: http://doi.acm.org/10.1145/2379776.2379787

[13] C. D. Nguyen, S. Miles, A. Perini, P. Tonella, M. Harman, and M. Luck, "Evolutionary testing of autonomous software agents," *Autonomous Agents and Multi-Agent Systems*, vol. 25, no. 2, pp. 260–283, Sep. 2012. [Online]. Available: https://doi.org/10.1007/s10458-011-9175-4

[14] R. B. Abdessalem, S. Nejati, L. C. Briand, and T. Stifter, "Testing Vision-based Control Systems Using Learnable Evolutionary Algorithms," in *Proceedings of the 40th International Conference on Software Engineering*, ser. ICSE '18. New York, NY, USA: ACM, 2018, pp. 1016–1026, event-place: Gothenburg, Sweden. [Online]. Available: http://doi.acm.org/10.1145/3180155.3180160

[15] M. Lindvall, A. Porter, G. Magnusson, and C. Schulze, "Metamorphic Model-Based Testing of Autonomous Systems," in *2017 IEEE/ACM 2nd International Workshop on Metamorphic Testing (MET)*, May 2017, pp. 35–41.

[16] C. Menghi, S. Nejati, K. Gaaloul, and L. Briand, "Generating Automated and Online Test Oracles for Simulink Models with Continuous and Uncertain Behaviors," *arXiv [cs]*, Mar. 2019. [Online]. Available: http://arxiv.org/abs/1903.03399

[17] "What is OSMOSIS? | Open-Source Material fOr Safety assessment of Intelligent Systems." [Online]. Available: https://osmosis.gitlab.io/ (Accessed 2019-07-25).