Metamorphic Testing of Autonomous Vehicles: a Case Study on Simulink

Pablo Valle
Mondragon University
Mondragon, Spain
pablo.valle@alumni.mondragon.edu

***Abstract*—Autonomous Vehicles (AVs) will revolutionize the way people travel by car. However, in order to deploy autonomous vehicles, effective testing techniques are required. The driving quality of an AV should definitely be considered when testing such systems. However, as in other complex systems, determining the outcome of a test in the driving quality on an AV can be extremely complex. To solve this issue, in this paper we explore the application of Quality-of-Service (QoS) aware metamorphic testing to test AVs modeled in MATLAB/Simulink, one of the predominant modeling tools in the market. We first defined a set of QoS measures applied to AVs by considering as input a recent study. With them, we define metamorphic relations. Lastly we assess the approach in an AV modeled in Simulink by using mutation testing. The results suggests that our approach is effective at detecting faults.**

I. Introduction

Autonomous Vehicles (AVs) are complex Cyber-Physical Systems (CPS) whose function is to transport passengers safely from a point to another while providing the best Quality of Service (QoS) as possible to its users. While there are certain AV properties that can be easily tested (e.g., whether a car has crashed against another), it is not always easy to determine which the test outcome should be, especially those related to QoS measures. For instance, determining the time required by a car to transport a person from a place to another can be extremely complex, making it difficult to catalogue a test as a “PASS” or as a “FAIL”. The resulting inability to determine whether a test outcome is correct or not is known as the *oracle problem* [3]. Subsequently, a manual assessment by the test engineer is often required in order to determine the outcome of a test, something that is costly.

Metamorphic Testing (MT) [6], [19] alleviates the oracle problem by adopting a singular approach to software testing: Instead of verifying the correctness of each execution, the relationships between the inputs and outputs of different executions are tested, called Metamorphic Relations (MR). Metamorphic testing has been used in many domains, such as machine learning applications, web services, computer graphics, and compilers [7], [18]. This technique has also been successfully applied in the domain of CPSs, such as for testing wireless sensor networks [5], autonomous drones [14], self-driving cars [20] or elevators [2]. However, to the best of our knowledge, the application of metamorphic testing has never been applied to testing AVs modeled in Simulink, the predominant CPSs modeling tool. Specifically, the main contributions of this paper can be summarized as follows:

1) We apply MT to testing AVs at system level using Simulink. Simulink is the predominant CPSs modeling tools and widely used in the automotive industry. To the best of our knowledge, this is the first approach where MT is used in the context of Simulink models. In addition, the application of MT at system level would allow detecting more significant safe violations [10], such as those caused by vehicle dynamics or uncertainties in the environment (e.g., ice on the road).
2) Similar to [2], our approach uses performance (QoS metrics) as a proxy to detect functional errors. These QoS metrics have been defined by analysing the paper by Jahangirova et al. [12]. We demonstrate the effectiveness of our method using mutation testing.
3) We make our method, case study and experimental material, available for replication purposes [21].

II. Background and related work

MT aims at detecting bugs by looking at the relationship among inputs and outputs of two or more executions of the program under test, called Metamorphic Relations (MRs). For example, consider an autonomous vehicle getting the following function: time(go(A,B)), which indicates the travel time from point A to point B by the shortest path. Checking if the output of the system is correct for two random input points would be difficult: this is an instance of the oracle problem. The order of the parameters should not influence the result (unless uncertain occurrences appear, e.g., an obstacle when traveling from A to B, but not from B to A). This could be expressed as the following MR time(go(A,B))==time(go(B,A)). In this relation, (A,B) is the source test case and (B,A), created by switching the inputs, is the follow-up test case. Every metamorphic relation could be instantiated into one or more metamorphic test by using specific input values and checking if the relation holds. If the relation is violated, the metamorphic test is said to have failed.

Simulink is the predominant CPSs modeling tool [8], especially in the automotive industry. Automated code generation compliant with the AUTOSAR standard is one of the reasons for this. Its wide usage has attracted the attention of researchers from the testing community to research on testing methods using Simulink [4], [11], [16], [23]. Several testing techniques have been proposed for Simulink models, including test generation [17], test selection [1] or fault localization [9], [15]. Unlike these approaches, we propose the usage of MT to alleviate the test oracle problem in an AV case study modeled in Simulink. MT has been used to testing AVs. There are many successful studies on metamorphic testing of driveless vehicles [20], [22], [24], [25]. Their focus is on testing the controller, which is implemented through a neural network. Unlike these approaches, our technique focuses on testing AVs at system level and modeled in Simulink. To the best of our knowledge, this is the first paper that uses MT to test AVs at system level using Simulink.

III. Aproach

We assessed several QoS metrics among the ones typically used in the domain in order to evaluate passenger experience. To do that, we carefully analysed the metrics proposed by Jahangirova et al. to develop test oracles for AVs [12]. These are the specified QoS metrics we selected for our MRs:

- **Time to destination (TD):** Time required for a vehicle to reach its destination. This is the relation between the speed and the distance to be driven.
- **Trajectory Offset (TO):**Trajectory offset is the difference of the Lateral Position (LP) from the start to the end of the driving task.

A test input in our case is composed by four inputs: (1) P_A and (2) P_B relate to the initial and destination points of the ego car in the map, (3) SP is the nominal speed and (4) OB is the number of obstacles that the car will have in its way during the simulation. We implemented a script that takes this as inputs and automatically executes a test, note that environment and road conditions are the same for all tests. As a test output, the above-mentioned QoS metrics are returned once the test case finishes its execution.

Following these QoS metrics and the test input we propose different relations for each of the metrics. For each Metamorphic Relation a Metamorphic Relation Input Pattern (MRIP) has been defined, which describes an input relation between the source and a follow-up test case exploited in different MRs. Specifically, we define the following MRIPs:

MRIP$_1$ – Increasing the nominal speed: This pattern represents those relations where the follow-up test case is constructed by increasing the nominal speed. When this happens the time to destination (TD) of the follow-up test case should be similar or lower to the source test case, represented as follows:

$$TD(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP_f, OB)) \lesssim TD(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP_s, OB)) \quad (\mathtt{MR1}_{\mathtt{TD}})$$

We can also define a similar relation for the Trajectory Offset (TO). When the speed is increased the TO might increase because the car becomes more difficult to be controlled, having the following MR:

$$TO(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP_f, OB)) \gtrsim TO(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP_s, OB)) \quad (\mathtt{MR1}_{\mathtt{S}})$$

MRIP$_2$ – Including an obstacle in the path: In this pattern, the follow-up test case is constructed with an extra obstacle inside the vehicle's path. When this happens the Time to Destination should increase, having the following MR:

$$TD(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP, OB_f)) \gtrsim TD(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP, OB_s)) \quad (\mathtt{MR2}_{\mathtt{TD}})$$

Conversely, the TO metric shall not change because the object detection implementation of our AV under test makes the ego-car to stop. Thus, the MR is defined as follows:

$$TO(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP, OB_f)) \simeq TO(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP, OB_s)) \quad (\mathtt{MR2}_{\mathtt{S}})$$

MRIP$_3$ – Swapping initial and destination points: This pattern represents those MRs where the initial and destination positions are swapped in the test input. When this happens this should not affect neither the Time to Destination (TD) nor the Trajectory Offset (TO). This relations could be represented as follows:

$$TD(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP, OB)) \simeq TD(move(P_{\mathrm{B}}, P_{\mathrm{A}}, SP, OB)) \quad (\mathtt{MR3}_{\mathtt{TD}})$$

$$TO(move(P_{\mathrm{A}}, P_{\mathrm{B}}, SP, OB)) \simeq TO(move(P_{\mathrm{B}}, P_{\mathrm{A}}, SP, OB)) \quad (\mathtt{MR3}_{\mathtt{S}})$$

IV. Evaluation and results

We evaluated our approach in an AV modeled in MATLAB/Simulink. We randomly generated 20 source test cases. For each source test case, its follow-up test cases were generated based on their MRIP. To assess the fault revealing capability of the defined metamorphic relations, mutation testing was used as it has been demonstrated to be a good substitute of real faults [13]. We generated a total of 8 mutants. The amount of mutants was not large because each test takes a long test execution time. Notice that, however, the amount of mutants is similar to those used in other testing studies where Simulink models are used [1]. The case study and the experimental material is avaible for replication purposes [21].

Table I shows the obtained mutation scores for each MR, each MRIP and the combination of all MRs. By combining all the MRs we were able to detect all the mutants. MRIP1 had the lowest mutation score, killing 5 out of 8 mutants (i.e., 63% of mutants), whereas MRIP2 and MRIP3 killed 7 out of 8 mutants. As for the QoS metrics, the TD metric showed a higher mutation scores with MRIP2 and MRIP3, whereas TO obtained a higher score with the MRIP1. The results show that the combination of the defined six MRs were able of killing all the seeded faults. This suggests that the combination of multiple MRs can kill different types of faults.

TABLE I: Mutation scores of MRs.

Metamorphic Relation		Mutation Score		
MRIP1	$\mathrm{MR1_{TD}}$	50%	63%	100%
	$\mathrm{MR1_{S}}$	63%		
MRIP2	$\mathrm{MR2_{TD}}$	88%	88%	
	$\mathrm{MR2_{S}}$	75%		
MRIP3	$\mathrm{MR3_{TD}}$	88%	88%	
	$\mathrm{MR3_{S}}$	50%		

V. Conclusion

We have proposed a technique based on QoS-aware metamorphic testing applied to testing AVs modeled in Simulink. The experimental results suggest that the technique is effective at detecting faults.

References

[1] Aitor Arrieta, Shuai Wang, Urtzi Markiegi, Ainhoa Arruabarrena, Leire Etxeberria, and Goiuria Sagardui. Pareto efficient multi-objective black-box test case selection for simulation-based testing. *Information and Software Technology*, 114:137–154, 2019.

[2] Jon Ayerdi, Sergio Segura, Aitor Arrieta, Goiuria SagarduiMaite Arratibel, and Maite Arratibel. Qos-aware metamorphic testing: An elevation case study. In *2020 IEEE 31st International Symposium on Software Reliability Engineering (ISSRE)*, pages 104–114. IEEE, 2020.

[3] Earl T Barr, Mark Harman, Phil McMinn, Muzammil Shahbaz, and Shin Yoo. The oracle problem in software testing: A survey. *IEEE transactions on software engineering*, 41(5):507–525, 2014.

[4] Angelo Brillout, Nannan He, Michele Mazzucchi, Daniel Kroening, Mitra Purandare, Philipp Rümmer, and Georg Weissenbacher. Mutation-based test case generation for simulink models. In *International Symposium on Formal Methods for Components and Objects*, pages 208–227. Springer, 2009.

[5] WK Chan, Tsong Y Chen, Shing Chi Cheung, TH Tse, and Zhenyu Zhang. Towards the testing of power-aware software applications for wireless sensor networks. In *International Conference on Reliable Software Technologies*, pages 84–99. Springer, 2007.

[6] T. Y. Chen, S. C. Cheung, and S. M. Yiu. Metamorphic testing: A new approach for generating next test cases. Technical report, Technical Report HKUST-CS98-01, Department of Computer Science, The Hong Kong University of Science and Technology, 1998.

[7] Tsong Yueh Chen, Fei-Ching Kuo, Huai Liu, Pak-Lok Poon, Dave Towey, T. H. Tse, and Zhi Quan Zhou. Metamorphic testing: A review of challenges and opportunities. *ACM Computing Surveys*, 51(1):4:1–4:27, January 2018.

[8] Yanja Dajsuren, Mark GJ Van Den Brand, Alexander Serebrenik, and Serguei Roubtsov. Simulink models are also software: Modularity assessment. In *Proceedings of the 9th international ACM Sigsoft conference on Quality of software architectures*, pages 99–106, 2013.

[9] Safa Aloui Dkhil, Mohamed Taha Bennani, Manel Tekaya, and Houda Ben Attia Sethom. Sequence mining and property verification for fault-localization in simulink models. In *International Conference on Dependability and Complex Systems*, pages 1–10. Springer, 2020.

[10] Fitash Ul Haq, Donghwan Shin, Shiva Nejati, and Lionel C Briand. Comparing offline and online testing of deep neural networks: An autonomous car case study. In *2020 IEEE 13th International Conference on Software Testing, Validation and Verification (ICST)*, pages 85–95. IEEE, 2020.

[11] Nannan He, Philipp Rümmer, and Daniel Kroening. Test-case generation for embedded simulink via formal concept analysis. In *Proceedings of the 48th Design Automation Conference*, pages 224–229, 2011.

[12] Gunel Jahangirova, Andrea Stocco, and Paolo Tonella. Quality metrics and oracles for autonomous vehicles testing. In *2021 IEEE 14th International Conference on Software Testing, Validation and Verification (ICST)*. IEEE, 2021.

[13] René Just, Darioush Jalali, Laura Inozemtseva, Michael D Ernst, Reid Holmes, and Gordon Fraser. Are mutants a valid substitute for real faults in software testing? In *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering*, pages 654–665, 2014.

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

[15] Bing Liu, Lucia, Shiva Nejati, Lionel C Briand, and Thomas Bruckmann. Simulink fault localization: an iterative statistical debugging approach. *Software Testing, Verification and Reliability*, 26(6):431–459, 2016.

[16] Bing Liu, Shiva Nejati, Lionel C Briand, et al. Improving fault localization for simulink models using search-based testing and prediction models. In *2017 IEEE 24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*, pages 359–370. IEEE, 2017.

[17] Reza Matinnejad, Shiva Nejati, Lionel C Briand, and Thomas Bruckmann. Test generation and test prioritization for simulink models with dynamic behavior. *IEEE Transactions on Software Engineering*, 45(9):919–944, 2018.

[18] S. Segura, G. Fraser, A. Sanchez, and A. Ruiz-Cortés. A survey on metamorphic testing. *IEEE Transactions on Software Engineering*, 42(9):805–824, Sept 2016.

[19] S. Segura, D. Towey, Z.Q. Zhou, and T.Y. Chen. Metamorphic testing: Testing the untestable. *IEEE Software*, 37(3):46–53, 2020.

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

[21] Pablo Valle and Aitor Arrieta. Experimental material: https://github.com/pablovalle/autonomous-vehicle, 2021.

[22] Xiaoyuan Xie, Joshua WK Ho, Christian Murphy, Gail Kaiser, Baowen Xu, and Tsong Yueh Chen. Testing and validating machine learning classifiers by metamorphic testing. *Journal of Systems and Software*, 84(4):544–558, 2011.

[23] Yuan Zhan and John A Clark. Search-based mutation testing for simulink models. In *Proceedings of the 7th annual conference on Genetic and evolutionary computation*, pages 1061–1068, 2005.

[24] Mengshi Zhang, Yuqun Zhang, Lingming Zhang, Cong Liu, and Sarfraz Khurshid. Deeproad: Gan-based metamorphic testing and input validation framework for autonomous driving systems. In *2018 33rd IEEE/ACM International Conference on Automated Software Engineering (ASE)*, pages 132–142. IEEE, 2018.

[25] Zhi Q Zhou and Liqun Sun. Metamorphic testing of driverless cars. 2019.