
Proceedings Paper:
Proma, Nawshin Mannan and Alexander, Rob (2023) Systematic Situation Coverage versus Random Situation Coverage for Safety Testing in an Autonomous Car Simulation. In: LADC 2023: 12th Latin-American Symposium on Dependable and Secure Computing. LADC 2023: 12th Latin-American Symposium on Dependable and Secure Computing, 16-20 Oct 2023 ACM, BOL, pp. 208-213.

https://doi.org/10.1145/3615366.3625077

Systematic Situation Coverage versus Random Situation Coverage for Safety Testing in an Autonomous Car Simulation

Nawshin Mannan Proma
nawshinmannan.proma@york.ac.uk
University of York
Heslington, York, UK

Rob Alexander
rob.alexander@york.ac.uk
University of York
Heslington, York, UK

ABSTRACT

Autonomous vehicles (AV) have the potential to improve road transport, but faults in the autonomous driving software can result in serious accidents. To assess the safety of AV driving software, we need to consider the wide variety and diversity of situations that it may encounter. Explicit situation coverage has previously been presented, but its usefulness has received a little empirical scrutiny. In this study, we evaluate a situation coverage based safety testing approach by comparing the performance of random and situation coverage-based test generation in terms of its ability to detect seeded faults in our ego AV at a road intersection under diverse environmental conditions. Our results suggest that this implementation of situation coverage, at least, does not provide an advantage over random generation.

CCS CONCEPTS

• **Software Engineering** ; • **Computer systems organization** → *Autonomous System*; Safety Engineering;

KEYWORDS

Safety, Autonomous car, Testing, Situation coverage

ACM Reference Format:
Nawshin Mannan Proma and Rob Alexander. 2023. Systematic Situation Coverage versus Random Situation Coverage for Safety Testing in an Autonomous Car Simulation. In *12th Latin-American Symposium on Dependable and Secure Computing (LADC 2023), October 16–18, 2023, La Paz, Bolivia.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3615366.3625077

1 INTRODUCTION

The revolution of autonomous technology and its potential benefits have driven the development of autonomous vehicles (AV) in the past decades. However, though autonomous vehicles have been developed, fully autonomous vehicles are still not safe enough for public use. The main obstacle comes from safety concerns. A 'safe' AV should be capable of following traffic laws and avoiding any road hazards independently and efficiently [15]. However, for a completely autonomous vehicle, human driver restrictions must be updated in accordance with AV safety concerns[10]. Moreover, the level of automation of a vehicle and its development can affect regulation-making [17], such as whether the human driver should monitor the surrounding environment throughout the journey or take control when it is an emergency [9].

In other words, AVs safety can be affected by numerous public and practical factors, including automation level definition, regulation-making, and vehicle types, road and traffic rules, and even weather conditions. As autonomous vehicles become more prevalent in public use, the potential for diverse failures increases. Assessing their safety is intricate due to the potential for widespread harm and the uncertain nature of interactions between pedestrians and autonomous vehicles.

To evaluate the safety of autonomous vehicles, we must pay close attention to the wide variety of situations they may encounter on the road. Because of this, testing for AV software differs from testing for conventional software. In simulation-based testing, coverage criteria [3] provide systematic ways to efficiently handle the vast number of possible inputs by searching the input space, which input should be selected, and when to stop testing. With the positive aspect of coverage criteria in mind, we consider coverage criteria-based testing for AVs' verification and validation (V&V). Recently, Scenario Coverage [19], Situation Coverage [2, 7], and Requirements Coverage [2] have been used as a coverage-based testing approach for AVs by many researchers.

However, it is hard to test AV software as AV can face a wide range of diverse external situations (obstacles, human) in an unstructured environment. To cover dynamic scenarios arising from the interaction of AV and its environment, a situation coverage approach is introduced by Alexander et al. [2]. Situation coverage [2] is a coverage criterion adapted to the testing of autonomous system. The fundamental idea is to identify potential environmental factors the system may encounter, to determine how they can change, and to make sure that both individual factors and their combinations are tested. Situation coverage, in its simplest form, is a measurement of the percentage of some potential circumstances that have been put to the test by some test set. Like other coverage criterion, situation coverage can be used to evaluate the adequacy of a test set and to guide automated test generation. For example, consider a scenario in which two people carrying a huge transparent glass and a courier robot both travel through the same hallway at the same time. With the use of its object detecting algorithm, the courier robot can identify the two people. However, because the glass is transparent, the robot might see it as a space between two people and attempt to cross it, colliding with it in the process. This

LADC/SAFELIFE '23, October 18, 2023, La Paz, Bolivia

ACM ISBN 979-8-4007-0844-2/23/10…$15.00
https://doi.org/10.1145/3615366.3625077

type of external circumstance may not even be addressed if the aforementioned autonomous system has been tested using a system coverage technique because it may not have been designed to deal with that by system component. Situation coverage is a promising coverage criterion, but more empirical research is needed to assess it.

There is a little existing work on situation coverage but only in narrow domains. For example, Zendel et.al.'s VITRO approach [20] only concerns itself with static scene and non-interactive videos for testing computer vision. It uses low geometric discrepancy sampling to make tractable subsets of situations but the published work doesn't rigorously evaluate its testing power. Andrew et. al. [5], [6], [4] explore several coverage criteria over their situation spaces-some of their criteria make tractable test sets but they perform no evaluation of actual fault finding power. Lesage and Alexander [16] established the SASSI approach, which specifies general guidelines for employing heuristic search to verify the safety of a cobot system. SASSI method is based on situation coverage approach by Alexander et al. [2] that propose the safety centric coverage metrics based on the observed situation in an industrial environment. However, although SASSI is ostensibly a situation coverage approach, the empirical work in [16] relies on a heuristic search that is guided not by situation coverage but by a measure of safety problems ("safety artefacts", in their terms) encountered. It will thus be hard to argue, in a safety case, that the results of the SASSI method give us any certain degree of confidence in the thoroughness of the testing. (This is likely to be true of any testing technique that use a measure of "failure badness" or similar as its fitness function).

Hawkins et.al [11] and Tahir and Alexander [18] evaluated situation coverage approach that is "Can a situation generation testing method based on situation coverage approach outperform a randomly generation one in terms of fault detection?" . Though the task environment of [11] and [18] were different with different seeded faults, they got almost same result. They concluded that the situation coverage does not achieve significant benefits over random situation generation but coverage directed generation does cover a greater proportion of the situation space in the same amount of compute time.

In this paper, we will use slightly different fault set on Tahir et.al. [18] approach and test whether situation coverage based search approach does add additional value to random generation or not. From Tahir's work, it appeared that situation coverage was *less* effective than random testing in finding seeded faults. However, Tahir's seeded faults were rather coarse-grained and perhaps thus quite easy to find. Our conjecture in this paper is that situation coverage testing will do better than random testing when the seeded faults are more difficult to find.

2 DEFINING SYSTEMATIC SITUATION COVERAGE FOR SAFETY TESTING

Situation coverage is a test coverage criterion that considers both the external and internal situations in which the autonomous system has been tested. When we claim that the tests are sufficient, it means that the system has been thoroughly tested across a wide range of situations. By systematizing this process and incorporating actual situation coverage metrics, we can enhance and improve the process, allowing for more accurate and valid claims about the adequacy of testing efforts [2].

Taking an illustration from a previous report by one of the authors [2] *"For instance, systematic situation coverage may create a straightforward criterion that provides us with a manageable space to assess the safety of AV. The requirement could be that the road map contains all potential intersections and that there are car,bike,HGV and pedestrians present with a variety of sorts and behavioral characteristics. We can now provide the requirement that "the ego AV must interact with each entity type at each junction shape," giving us a total of 15*4=60 test cases. In a situation set, this kind of safety coverage metrics may help us uncover flaws that other coverage criteria could have overlooked"* .

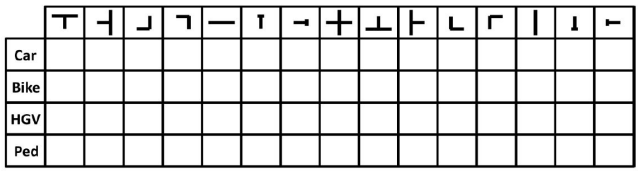

Figure 1: Example of situation coverage [2]

It is possible to assess situation coverage measures through empirical evaluation, although relying solely on this approach may not be sufficient to substantiate safety-critical claims. One way to achieve this is by introducing faults into the AV and then evaluating the effectiveness of situation coverage guided testing in identifying the corresponding system behavior resulting from these faults [2].

3 OVERVIEW OF SITCOV AV TESTING FRAMEWORK

To achieve situation coverage, Tahir and Alexander [18] created an automatic test suite generator called the SitCov AV-testing framework. This framework progressively generates tests, using situation coverage metrics to determine which situations have already been covered and how frequently each situation has been generated. Based on this information, the test suite generator generates the next set of situations in order to cover more of the situation space. It also aims to achieve a close-to-uniform distribution in cases where situations are repeated, provided that all situations have been covered at least once.

3.1 Situation hyperspace

Tahir and Alexander [18] refers to the surrounding environment around the ego AV as situation hyperspace. This situation hyperspace has been built methodically so that the SitCov AV-testing framework can systematically navigate through it to generate interesting and challenging situations for our ego AV using situation coverage-based generation to ensure the SitCov AV-testing framework executes good coverage of the situation hyperspace. The SitCov AV-testing framework employs a selection process that involves choosing situation elements from various sources, including environmental conditions and intersection axes. These elements

are then combined to create a discrete situation in which the autonomous vehicle (AV) simulation is executed within the CARLA platform [18].

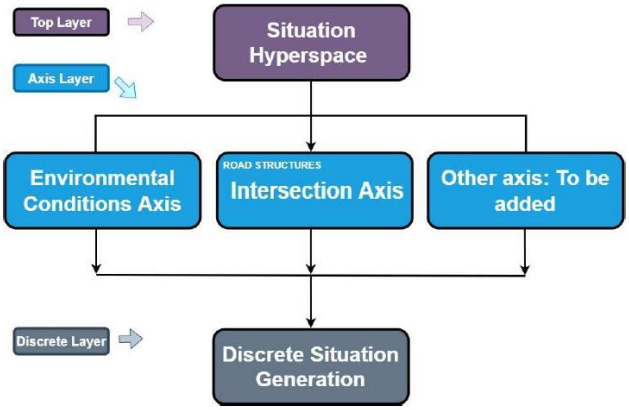

Figure 2: Situation Hyperspace (from [18])

3.2 SitCov AV testing Approach

The process of generating discrete situations from the situation hyperspace is driven by SitCov AV testing approach. This is accomplished by keeping track of how frequently each bin of the situation elements from the environmental conditions and intersection axes is utilized during the generation process. The counts of each bin represent the number of times it has been selected for a discrete situation generation simulation run. In simpler terms, bins of situation elements (e.g., precipitation in environmental conditions or intersection situation labels) that have been generated fewer times than others have a higher probability of being chosen by the SitCov AV-testing approach for the next simulation run. The block diagram of the SitCov AV-testing illustrates the visible steps involved in the process in figure 2.

3.3 Reproducing Tahir's Results

To reproduce SitCov AV-testing approach, we utilized Scenario Runner [8], which is an API integrated with CARLA. This combination allowed us to develop specific scenarios for testing autonomous vehicles. Our focus was on generating intersection scenarios as Tahir and Alexander [18] did, so we modified the intersection class in Scenario Runner to support AV-testing test suites. By leveraging Python, we reproduced situation hyperspace using dictionaries and lists to generate situation coverage-based scenarios on a T-intersection, see figure 3.

To assess the effectiveness of SitCov AV-testing approach, Tahir and Alexander [18] chose a modular pipeline autonomous driving (AD) algorithm for the ego AV during simulation runs. The modular pipeline for autonomous driving contains the following main phases: perception, local planning, and continuous control are the first three.

In CARLA, the local planner stage was already developed to determine the best route between two points on the map using the

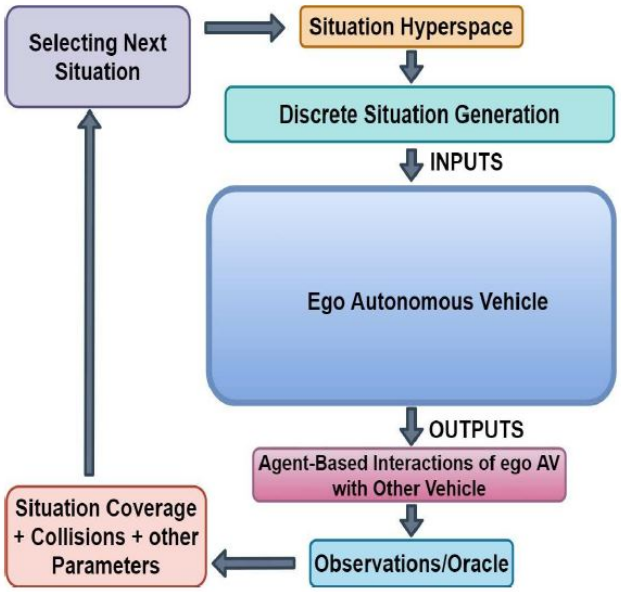

Figure 3: SitCov AV Testing Approach (from [18])

Figure 4: SitCov AV-Testing Approach in CARLA and Scenario Runner

roads and observing traffic laws. The continuous controller stage, which Tahir used directly for the ego AV and other vehicle (OV) controllers, is a PID controller for lateral and longitudinal control of the ego vehicle. This controller was also already developed in CARLA.

The perception stage is a further stage that the ego AV employs for its autonomous driving whereas the OV does not. In this instance, the ego AV's perception sensor is limited to the camera. To detect the approaching OV, locate its position in the dash-cam feed (images) of the ego AV, and apply emergency brakes (AEBs) if the OV is too close, Tahir uses a very deep Single-Shot multi-box Detection (SSD) Convolutional Neural Network (CNN) pre-trained on 350,000 images of the MS COCO data set, the SSD mobile net [13], [1], [12]. This pre-trained SDD mobilenet CNN is implemented using the Tensorflow Object Detection API [14].

4 RE-EVALUATING TAHIR'S METHOD WITH NARROWER FAULTS

The perception stage of the ego AV in the CARLA environment is influenced by three key parameters. These parameters relate to how the AV processes images captured by the ego dash-cam. By modifying them, we can seed faults into the AV's software. The three parameters are as follows:

(1) *Probability Detection Threshold for Object Detection:* This threshold determines the minimum probability value outputted by the SSD Mobilenet CNN for detecting a car in the images received from the ego AV's dash-cam in CARLA. As we focus on vehicle to vehicle (V2V) pairwise testing with an OV, our concern lies specifically with the probability of detecting cars in the received images. Setting this parameter for probability detection is crucial as it determines the minimum probability value that the ego AV considers as a valid detection of an OV in the image. Selecting an excessively high value could lead to missed detections and potential collisions in extreme weather conditions like heavy rain or fog, where the probability of detecting the OV may be lower even if it is present in front of our ego AV. Our seeded **fault-1** is setting the Probability of Object Detection Threshold to a relatively lower value of **0.80**. In contrast, Tahir's [18] ego AV required a higher probability threshold of 0.95 to detect the OV in front of it.

(2) *Centering Limits Parameters:* These parameters evaluate whether the OV detected by the SSD Mobilenet CNN is positioned close to the center of the received image. If the detected OV is too far from the center, the activation of Automatic Emergency Brakes (AEBs) is withheld, even if the OV is in close proximity to the ego AV. Tahir [18] established very strict boundaries for the centering limits parameters, which means that AV will only classify an object in front of it as a potential danger if it is precisely positioned in the center of the image captured by the AV's dash-cam. This approach disregards the potential hazard even if the object is dangerously close to the AV. During our experiment, we seeded **fault-2** by setting values part-way between the default values and Tahir's fault-2 values.

(3) *Object detection distance threshold:* To determine the distance of the OV from the image, we employ a computer vision technique that analyzes the percentage of the image feed occupied by the detected OV. As the OV gets closer to the ego AV, a higher percentage of the ego AV's image will contain the OV. Once the percentage exceeds a certain threshold value, the ego AV initiates the application of Automatic Emergency Brakes (AEBs) to avoid a collision. In Tahir's [18] conducted experiment on seeded **fault-3** , he established a remarkably low threshold for the distance parameter in object detection. Consequently, his autonomous vehicle (AV) will only perceive an object in its surroundings as a potential hazard if it is within an extremely close proximity, specifically 1 meter away. This adjustment of the distance threshold parameter, to a distance of 1.5 meters in our experiment, means that the AV will solely consider objects within this minimal range as potential dangers.

To gain a clearer understanding of the faults set, please refer to table 1.

5 RESULT: SITUATION COVERAGE VERSUS RANDOM GENERATION

In the course of Tahir's experiment involving a hard fault set, it was observed that the fault revealing capability of a random generation method exhibited better performance in contrast to a situation coverage-based approach. Interestingly, when the system was operating under a fault-free condition, both methods yielded an equivalent number of failures. When three distinct fault sets were introduced, the random approach found a comparatively higher number of failures compared to the situation coverage approach [18].

In light of these findings, we proceeded to conduct two additional experiments: one without any faults and another under the condition of three narrower fault sets. The objective was to comprehensively assess and compare the performance of both approaches, with an underlying hypothesis that the situation coverage-based method would perform better. The results of these two experiments are presented herein.

5.1 Experiment 1 — Performance of the two situation generation methods with no faults seeded

In our first set of tests, we compare the results of our SitCov AV-testing framework's intersection situation generation to those of random generation when no faults have been seeded. We conducted three sets of runs for this experiment, and the figure below illustrates the intersection situation along with the distributions of failures for both the SitCov and random generation methods under a no-seeded fault condition.

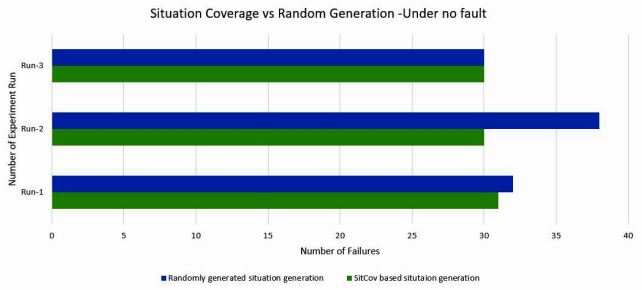

Figure 5: SitCov vs Random-No Fault Seeded

Based on Figure 5, we observe that the total number of failures is nearly similar for both the SitCov and random generation methods. In the second run of the experiment, the number of failures is significantly higher for the random generation method compared to SitCov. However, in runs 1 and 3, the failure numbers were approximately equal for both situation generation methods. This peculiar behavior of the random generation method is likely due to the uneven distribution of situations produced by the random generation process.

Table 1: Re-evaluating SitCov AV testing with narrower faults

Fault ID	Parameter	Normal Case Value	Tahir's Fault Value	Our Fault Value
\Fault-1	Probability of object detection threshold	0.70	0.95	0.80
\Fault-2	Centering limit parameters	0.02-0.98	0.3-0.6	0.2-0.8
\Fault-3	Object detection distance threshold	2.5	1	1.5

5.2 Experiment 2 — Performance of the two situation generation methods with faults seeded

In our second test set, we conducted a comparison between the results of our test cases, which incorporated slightly different seeded faults, and Tahir's test cases. Considering that our fault sets were designed with the assumption that the SitCov method would outperform randomly generated situations, the figure below illustrates that, contrary to our expectations, the randomly generated method demonstrated better fault revealing capabilities in both Tahir's and our fault sets.

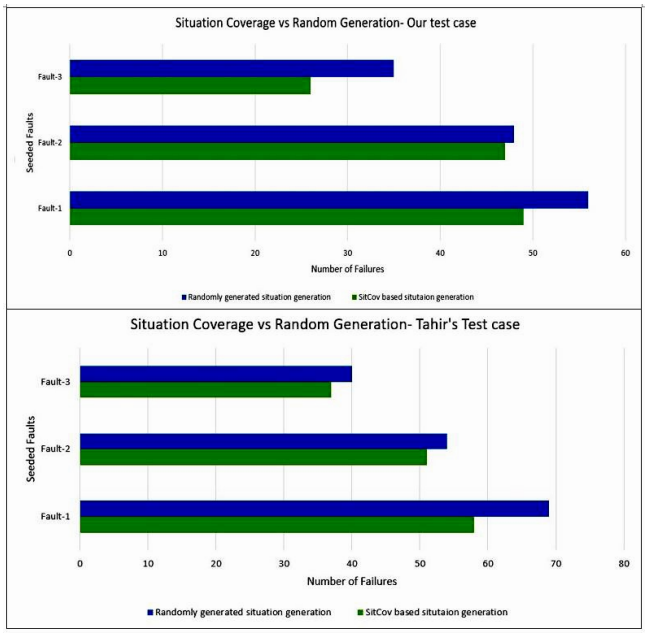

Figure 6: SitCov vs Random- Fault Seeded(Our test case vs Tahir's [18]

As seen in Figure 6, fault 1 is triggered the most in both test case even with relatively lower value in our case, which tells us that the parameter Probability of Object Detection Threshold, that we tinkered as a part of our fault 1 seeding, is extremely important for autonomous driving. Regarding Fault 2, which involves centering limits parameter, our experiments indicate that the fault revealing capabilities of our SitCov AV-testing framework versus random generation are comparable in both test cases. However, when it comes to Fault 3 in our case, an intriguing outcome arises in our SitCov case. The number of failures is surprisingly lower than the no-fault condition, with a count of 27. This peculiar behavior could be attributed to the utilization of a relatively simple algorithm and the presence of just one RGB-camera sensor on our ego AV.

6 CONCLUSION AND FURTHER STEPS

In conclusion, our research has examined the effectiveness of the SitCov AV testing framework in generating detecting seeded faults. Our observations indicate that SitCov did not perform well when compared to random generation in both our fault sets and Tahir's fault sets.

It may still be worth conducting further research on SitCov methods, particularly by expanding the situation hyperspace with additional axes. By incorporating more dynamic obstacles and scenarios, such as different types of roads, diverging and merging collisions, and the inclusion of pedestrians and cyclists, the situational complexity in AV testing can be enhanced. By expanding the situation hyperspace in these ways, it may be true that the SitCov AV testing approach has the potential to yield valuable insights and improve the overall effectiveness of AV testing. Future research should focus on incorporating these additional dimensions and conducting experiments to assess its performance.

ACKNOWLEDGMENTS

We would like to thank my colleague Victoria Hodge in the Institute for Safe Autonomy for her support throughout this study. Her assistance in reproducing the approach has enhanced the quality of this work.

We are grateful for the financial support provided to this project by funding from the European Union's EU Framework Programme for Research and Innovation Horizon 2020, under Grant Agreement No. 812788.

REFERENCES

[1] [n. d.]. https://cocodataset.org/
[2] Rob Alexander, Heather Rebecca Hawkins, and Andrew John Rae. 2015. Situation coverage–a coverage criterion for testing autonomous robots. (2015).
[3] Paul Ammann and Jeff Offutt. 2016. *Introduction to software testing*. Cambridge University Press.
[4] Anneliese Andrews, Mahmoud Abdelgawad, and Ahmed Gario. 2015. Active World Model for Testing Autonomous Systems Using CEFSM.. In *MoDeVVa@ MoDELS*. 1–10.
[5] Anneliese Andrews, Mahmoud Abdelgawad, and Ahmed Gario. 2015. Towards world model-based test generation in autonomous systems. In *2015 3rd International Conference on Model-Driven Engineering and Software Development (MODELSWARD)*. IEEE, 1–12.
[6] Anneliese Andrews, Mahmoud Abdelgawad, and Ahmed Gario. 2016. World model for testing autonomous systems using petri nets. In *2016 IEEE 17th International Symposium on High Assurance Systems Engineering (HASE)*. IEEE, 65–69.
[7] Aren A Babikian. 2020. Automated generation of test scenario models for the system-level safety assurance of autonomous vehicles. In *Proceedings of the 23rd ACM/IEEE International Conference on Model Driven Engineering Languages and Systems: Companion Proceedings*. 1–7.

[8] Carla-Simulator. [n. d.]. https://github.com/carla-simulator/scenario_runner
[9] Daniel J Fagnant and Kara Kockelman. 2015. Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations. *Transportation Research Part A: Policy and Practice* 77 (2015), 167–181.
[10] Francesca M Favarò, Nazanin Nader, Sky O Eurich, Michelle Tripp, and Naresh Varadaraju. 2017. Examining accident reports involving autonomous vehicles in California. *PLoS one* 12, 9 (2017), e0184952.
[11] Heather Hawkins and Rob Alexander. 2019. Situation Coverage Testing for a Simulated Autonomous Car–an Initial Case Study. *arXiv preprint arXiv:1911.06501* (2019).
[12] Andrew G Howard, Menglong Zhu, Bo Chen, Dmitry Kalenichenko, Weijun Wang, Tobias Weyand, Marco Andreetto, and Hartwig Adam. 2017. Mobilenets: Efficient convolutional neural networks for mobile vision applications. *arXiv preprint arXiv:1704.04861* (2017).
[13] Jonathan Huang, Vivek Rathod, Chen Sun, Menglong Zhu, Anoop Korattikara, Alireza Fathi, Ian Fischer, Zbigniew Wojna, Yang Song, Sergio Guadarrama, et al. 2017. Speed/accuracy trade-offs for modern convolutional object detectors. In *Proceedings of the IEEE conference on computer vision and pattern recognition*. 7310–7311.
[14] H. Kinsley. 2017. Object detection with tensorflow - self driving cars p.17. https://www.youtube.com/watch?v=UAXulqzn5Ps
[15] Philip Koopman and Michael Wagner. 2017. Autonomous Vehicle Safety: An Interdisciplinary Challenge. *IEEE Intelligent Transportation Systems Magazine* 9, 1 (2017), 90–96. https://doi.org/10.1109/MITS.2016.2583491
[16] Benjamin Lesage and Rob Alexander. 2021. SASSI: safety analysis using simulation-based situation coverage for cobot systems. In *Computer Safety, Reliability, and Security: 40th International Conference, SAFECOMP 2021, York, UK, September 8–10, 2021, Proceedings 40*. Springer, 195–209.
[17] Greig Mordue, Anders Yeung, and Fan Wu. 2020. The looming challenges of regulating high level autonomous vehicles. *Transportation research part A: policy and practice* 132 (2020), 174–187.
[18] Zaid Tahir and Rob Alexander. 2022. Intersection focused Situation Coverage-based Verification and Validation Framework for Autonomous Vehicles Implemented in CARLA. In *Modelling and Simulation for Autonomous Systems: 8th International Conference, MESAS 2021, Virtual Event, October 13–14, 2021, Revised Selected Papers*. Springer, 191–212.
[19] Simon Ulbrich, Till Menzel, Andreas Reschka, Fabian Schuldt, and Markus Maurer. 2015. Defining and substantiating the terms scene, situation, and scenario for automated driving. In *2015 IEEE 18th international conference on intelligent transportation systems*. IEEE, 982–988.
[20] Oliver Zendel, Wolfgang Herzner, and Markus Murschitz. 2013. Vitro-model based vision testing for robustness. In *IEEE ISR 2013*. IEEE, 1–6.

Received date tbc; revised date tbc; accepted date tbc