Simulation-based Fault Injection in Advanced Driver Assistance Systems Modelled in SUMO

Mehdi Maleki, Behrooz Sangchoolie

Dependable Transport Systems, RISE Research Institutes of Sweden, Borås, Sweden {mehdi.maleki, behrooz.sangchoolie}@ri.se

I. Introduction

Embedded electronic systems used in vehicles are becoming more exposed and thus vulnerable to different types of faults and cybersecurity attacks. Examples of these systems are advanced driver assistance systems (ADAS). Failures in these systems could have severe consequences. Therefore, these systems should be thoroughly evaluated during different stages of product development. An effective way of evaluating these systems is through the injection of faults and monitoring their impacts on these systems. Fault injection can be conducted either through the field tests or simulation-based tests. While conducting field tests could be costly and sometimes life-threatening [1], [2], simulation-based tests provide a wide range of advantages, such as adaptation of tests to a variety of traffic scenarios and avoiding the life-threatening situations.

In this paper, we present SUFI, a simulation-based fault injector that is capable of injecting faults into ADAS features (e.g., car-following and lane-changing) modelled in SUMO (simulation of urban mobility). Using SUFI, we model different faults and measure their impacts on the target system. The results of the fault injection experiments show the effectiveness of SUFI in revealing weaknesses of ADAS features modelled in SUMO when targeted by faults and attacks.

II. Experimental Setup

A. SUFI (SUmo-based Fault Injector)

Fig. 1 presents the architecture of SUFI that uses SUMO [3] to simulate the traffic and model the ADAS features. Python scripts are written to specify fault models (see §II-B), fault locations (see §II-E), fault injection time interval (see §II-D), and data logging commands. The Python scripts and SUMO are communicating with each other via TraCI (Traffic Control Interface) [4] during experiment runs.

B. Fault and Attack Models

SUFI is capable of modelling different types of faults and attacks such as, (i) **stuck-at-value** where the content of a location is stuck at a certain value. If the value is stuck permanently, it could be used to model manufacturing defects e.g., in sensors. However, if it is stuck temporarily, it could be used to model cybersecurity attacks such as the *replay* attack. And (ii) **single/double bit-flip** where up to two of the bits stored in a target location are flipped and is used to model cybersecurity attacks such as *corrupt message*.

SUFI is also capable of modelling *transient* and *semi-permanent* faults. The former is used to model faults that only appear in the system for one simulation time step or to model temporal cybersecurity attacks while the latter is used to model faults that remain in the system until the end of a simulation run such as complete blockage of a sensor.

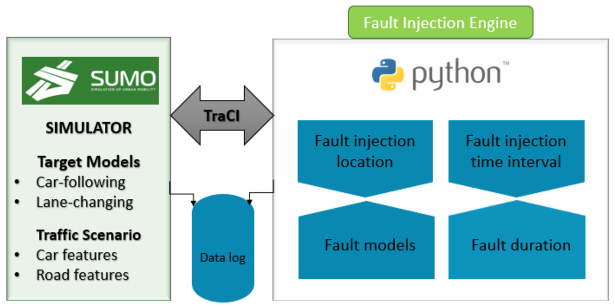

Fig. 1: Architecture of SUFI (SUmo-based Fault Injector)

C. Models Under Evaluation

This study evaluates the *longitudinal* and *lateral* behaviours of vehicles through the injection of faults and attacks into ACC (Adaptive Cruise Control) [5] and CACC (Cooperative Adaptive Cruise Control) [5], [6] *car-fallowing* models as well as LC2013 [7] *lane-changing* model.

D. Traffic Simulation Scenario and Time Interval

We use a three-lane road network scenario, where 10 vehicles are driving on the road during a simulation run. The road's speed limit is 36 m/s and its length is 750 m. We refer to the vehicle that is the target of our faults the *ego-vehicle*.

The total simulation time for the defined traffic scenario is 42 s and the interval between 11 s and 21 s is the fault injection time interval as it is very likely for the ego-vehicle to perform a lane-change within this period, facilitating the evaluation of the car-following and lane-changing models.

E. Fault Injection Locations

The vehicle behaviour could be affected by injection of faults in a variety of parameters used by the car-following and

This work was supported by VALU3S project, which has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876852. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Czech Republic, Germany, Ireland, Italy, Portugal, Spain, Sweden, Turkey.

TABLE I: Fault injection results.

Fault Model	Duration	Non-Effective	Crash	Negligible	Benign	Severe	Total
LC-assertive parameter (fault injection time interval: 11.0 s – 21.0 s)							
Stuck-at-value	Semi-permanent	9	0	597	98	1296	2000
	Transient	3	0	600	100	1297	2000
Single bit-flip	Semi-permanent	1220	20	12	2	26	1280
	Transient	1222	20	10	2	26	1280
Double bit-flip	Semi-permanent	1843	54	22	7	74	2000
	Transient	1842	56	22	5	75	2000
Reaction time parameter (fault injection time interval: 11.0 s – 21.0 s)							
Stuck-at-value	Semi-permanent	0	0	282	572	146	1000
	Transient	0	0	550	450	0	1000
LC-assertive + reaction time parameters (fault injection time interval: 11.0 s – 14.0 s)							
Stuck-at-value	Semi-permanent	0	0	2703	9697	5100	17500

lane-changing models. The effectiveness of these injections in manipulating the vehicle behaviour could also be tightly connected to the defined traffic scenario. In this paper, we target two parameters in SUMO. The first one is *LC-assertive* which shows the eagerness of the vehicle to perform a lane change and allows us to model sensor faults resulting in incorrect gap estimations. Here, LC is an acronym for Lane-Change. The second parameter is *reaction time* corresponding to the update time of the vehicle behaviour which allows us to model a delay attack.

F. Outcome Classifications

We classify the results in five groups by looking into the deceleration (braking rate) profiles of the vehicles; (i) *Non-effective*: the injected fault has no effects on the behaviour of the vehicles, (ii) *Negligible*: the recorded maximum deceleration is less than or equal to 0.78 m/s^2, (iii) *Benign*: the recorded maximum deceleration is between 0.78 m/s^2 and 5.0 m/s^2, (iv) *Severe*: the recorded maximum deceleration is greater than 5.0 m/s^2, some of which also result in car collisions, (v) *Crash*: the simulator crashes after the fault injection. Note that, the braking rate 0.78 m/s^2 is the maximum deceleration recorded in the golden run for the target time interval (see §II-D) and the maximum comfortable braking rate and the maximum emergency braking rate are defined as 5.0 m/s^2 [8] and 8.0 m/s^2 [9], respectively.

III. Experimental Results

Table I shows the results of the fault injection experiments conducted. The table shows a great number of benign and severe cases for when the stuck-at-value model is used. The single and double bit-flip models, on the contrary, are significantly less effective in causing severe results. The table also shows that in general, the difference between the results obtained for the transient and semi-permanent models are insignificant except for when the stuck-at-value is used when targeting *reaction time* parameter.

When analysing the result of fault injection experiment presented in Table I for the *LC-assertive* parameter, we learned that none of the faults injected during the period between 11.0 s and 14.0 s resulted in a severe case. Therefore, we decided to further investigate this time interval by exposing the system to faults in both the *LC-assertive* and *reaction time* parameters (see the last line of Table I). The obtained results show the effectiveness of these faults in revealing system weaknesses that could not be revealed when targeting a single location.

It is also worth mentioning that, the faults injected, in addition to affecting the speed profile of ego-vehicle, influences multiple other nearby vehicles. In fact, for the *LC-assertive*, the vehicle behind the ego-vehicle is the one flagging all the severe cases, whereas for the *reaction time*, the majority of the severe cases are flagged by the ego-vehicle.

IV. Conclusions and Implications

In this paper, we presented SUFI that allowed us to successfully reveal some of the weaknesses of ADAS features modelled in SUMO through fault injection experiments. The experiment results show that parameters such as the fault model, fault location and the duration in which a system is exposed to faults play an important role in revealing these weaknesses. As part of the future work, we plan to perform injections in other locations of the CACC and ACC models and to model other groups of faults/attacks. Besides, as physical features of the environment such as the weather condition are not considered in SUMO, we plan to use simulators such as CARLA [10] that provides us with modelling of such features.

References

[1] "Waymo Self-driving Car Crash," https://www.theverge.com/2018/5/4/17320936/waymo-self-driving-car-crash-arizona, accessed: 2021-04-23.

[2] "Uber Self-driving Car Test," https://en.wikipedia.org/wiki/Death_of_Elaine_Herzberg, accessed: 2021-04-23.

[3] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic traffic simulation using sumo," in *2018 21st Int. Conf. on Intelligent Transportation Systems (ITSC)*, 2018, pp. 2575–2582.

[4] "Traffic Control Interface," https://sumo.dlr.de/docs/TraCI.html, accessed: 2021-04-23.

[5] V. Milanés and S. E. Shladover, "Modeling cooperative and autonomous adaptive cruise control dynamic responses using experimental data," *Transportation Research Part C*, vol. 48, pp. 285–300, 2014.

[6] K. N. Porfyri, E. Mintsis, and E. Mitsakis, "Assessment of ACC and CACC systems using SUMO," *EPiC Series in Engineering*, vol. 2, 2018.

[7] J. Erdmann, "Lane-changing model in sumo," *Proceedings of the SUMO2014 modeling mobility with open data*, vol. 24, pp. 77–88, 2014.

[8] "SUMO Vehicle Type Parameter Defaults," https://sumo.dlr.de/docs/Vehicle_Type_Parameter_Defaults.html, accessed: 2021-04-23.

[9] P. Greibe, "Braking distance, friction and behaviour," *Trafitec, Scion-DTU*, 2007.

[10] "CARLA Open-source simulator for autonomous driving research," https://carla.org/, accessed: 2020-12-01.