Limitations of HIL Test Architectures for Car2X Communication Devices and Applications

Christina Obermaier
christina.obermaier@thi.de
Technische Hochschule Ingolstadt
Ingolstadt, Germany

Raphael Riebl
raphael.riebl@thi.de
Technische Hochschule Ingolstadt
Ingolstadt, Germany

Ali H. Al-Bayati
alihmohd@dmu.ac.uk
School of Computer Science and
Informatics, De Montfort University
Leicester, United Kingdom

Christian Facchi
christian.facchi@thi.de
Technische Hochschule Ingolstadt
Ingolstadt, Germany

Sarmadullah Khan
sarmadullah.khan@dmu.ac.uk
School of Computer Science and
Informatics, De Montfort University
Leicester, United Kingdom

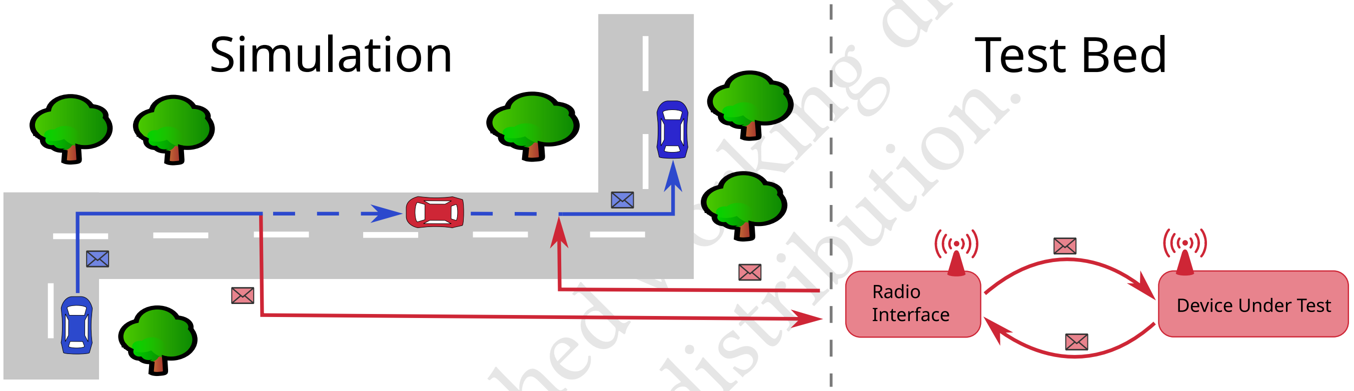


Figure 1: Schematic overview of a *Vehicle-to-Everything* (V2X) *Hardware in the Loop* (HIL) simulation.

ABSTRACT

Evaluating vehicle software and hardware using *Hardware in the Loop* (HIL) simulation is a very common process in current vehicle manufacturing. However, the more complex the vehicle's environmental awareness becomes, the more complex the HIL simulation framework has to become. With the introduction of *Vehicle-to-Everything* (V2X) communication, the environmental awareness of traffic participants expands tremendously. Yet, appropriate tools for evaluating *Electronic Control Units* (ECUs) with a high level of environmental awareness are lacking. Considering scenarios with more than a handful of vehicles, current HIL simulation frameworks are not capable of simulating these scenarios in real time. Hence, the state-of-the-art testing approach is to provide a non-reactive environment for the validation of V2X ECUs. This paper addresses the question, if and to which extent such a non-reactive approach is sufficient for validating complex V2X based applications.

KEYWORDS

V2X communication, validation, hardware in the loop, intelligent transport systems

ACM Reference Format:
Christina Obermaier, Raphael Riebl, Ali H. Al-Bayati, Christian Facchi, and Sarmadullah Khan. 2019. Limitations of HIL Test Architectures for Car2X Communication Devices and Applications. In *Kaiserslautern '19: ACM Computer Science in Cars, October 08, 2019, Kaiserslautern, Germany.* ACM, New York, NY, USA, 9 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

Advanced Driver Assistance Systems (ADASs) take over increasing responsibility in road transport. They inform the driver about critical situations or even take over the control of the vehicle in certain situations entirely [30]. For example, an automatic brake assistant tries to avoid crashes with obstacles ahead by performing an emergency brake if necessary. Obviously, such systems highly rely on the environmental awareness of the vehicle.

---

*This work has been conducted in the project SAFIR funded by the German Ministery of Education and Research based on the funding line FH-Impuls, 13FH71031A.

ADASs are only able to react upon a situation if they are able to detect and validate an incident by fusing information gathered from the vehicle's sensors. Current series cars are already able to capture their environment quite extensively by camera, radar or laser sensors [39]. However, these sensors share one substantial problem: They require a line of sight to detect a possible critical situation [33]. This means, that many critical situations may be missed by vehicles because of an insufficient field of view. *Car-to-Everything* (C2X) communication can help to tackle this field of view problem and enhance the environmental awareness of vehicles significantly.

C2X (or *Vehicle-to-Everything* (V2X) more general) communication describes a communication system, which is meant to interconnect road participants like vehicles and road infrastructure. In Europe, *European Telecommunications Standards Institute* (ETSI)'s *Intelligent Transport System* (ITS) group specifies how to establish such communication based on a *Vehicular Ad Hoc Network* (VANET) [15]. Moreover, so-called "Day One" applications such as *Cooperative Awareness* (CA) and *Decentralized Environmental Notification* (DEN) are specified as well [16, 17]. As these applications aim to enhance traffic safety and traffic flow, it is necessary to validate the corresponding software and hardware implementations before deploying them in the field. However, conducting field tests for every validation aspect is enormously time consuming and prohibitively expensive and reproducibility is limited. Moreover, some features are linked to dangerous driving situations and can thus hardly be mimicked in real field tests safely [22]. Therefore, it is inevitable to provide *Hardware in the Loop* (HIL) simulations for testing these applications and their underlying hardware.

This paper is organised in the following way: Section 2 presents insights of current HIL frameworks in various areas of application. Moreover, state-of-the-art testing methods for V2X communication in general are evaluated. Section 3 moves on to analyse the characteristics and limitations of various HIL approaches for V2X components. Specifically, the HIL modes "open-loop", "closed-loop" and "semi-closed-loop" are investigated. Section 4 discusses the impact of aforementioned limitations. Finally, Section 5 concludes the work and presents future research directions to mitigate the discussed problems.

2 RELATED WORK

HIL simulation and HIL test beds in general are a very variadic research field. Because of their wide area of application, many solutions for different purposes exist [1]. However, in the field of V2X communication, there is still a lack of comprehensive and powerful test beds for testing active V2X components like *Electronic Control Units* (ECUs).

2.1 HIL Simulation

When speaking about HIL simulation one has to consider that seldom a formalized approach in designing such a system exists. Most times, they are created to fit very specific purposes. Nevertheless, these systems share some common design ideas and basic principles as they all have to cope with similar problems [1].

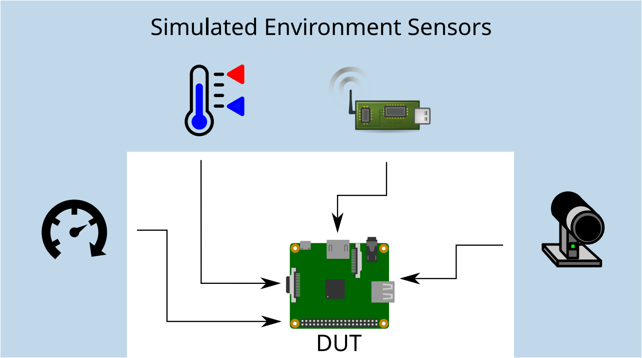


Figure 2: Architecture of a HIL test bed

Figure 2 shows a generic HIL system: A *Device Under Test* (DUT) is placed in a simulated environment which provides at least one input source for the DUT. Depending on the area of application, the simulated environment may be defined in more detail. For example, Karunagaran et al. describe how to validate an insulin pump [23]. They use a time and event driven simulation system which imitates the actual environment of the insulin pump. For this purpose they have established sensor interfaces which are evaluated by the pump's actuators. Hence, this system enables hardware tests of a health critical device without the risk to harm anyone during the test run. Even more, fault injection tests can be executed without endangering a real patient.

HIL simulation is not only used in life critical areas but also where affordable and manageable system-wide tests for hardware and software are needed. Hematian et al. showed how to create a test bed for the assessment of smart meter traffic over *Long Term Evolution* (LTE) [20, 21]. They simulated a smart meter environment and sent their traffic over LTE using *Software Defined Radio* (SDR) devices. This allowed them to evaluate the LTE traffic in real time.

Baracos et al. described back in 2001 the increasing importance of HIL testing in the automotive sector [2]. They showed that it is necessary to have deterministic, real-time capable simulation systems to ensure realistic and usable results. Jia and Cebon showed that vehicle systems and their software have grown in complexity massively since that time [22]. Therefore, testing an ECU in a well-defined environment is crucial to avoid fatal integration errors.

Even though a shear unlimited variety of possible test beds exists, they share a common aim: Providing a well-defined environment that cannot be distinguished by the DUT from the real world. This way it can be safely assumed that the DUT behaves identical in both, the HIL and the final environment [1].

Test beds are employed whenever extensive field tests would be too expensive or too dangerous in case of safety-critical applications [36]. Also, they allow to evaluate the system under real-time conditions before putting its hardware into the field or production [1]. However, with increasing complexity of software, the HIL test beds became more complex as well. This leads to a quite extensive issue: In some cases, current computing systems and software architectures are not fast enough to execute the model in real time any more. Especially in the area of V2X testing solutions, computing time increases significantly [4, 26, 28].

2.2 Simulation-based Testing of V2X Systems

When developing new V2X applications, its not the first step to install them on an ECU immediately. Speth et al. showed the advantages of *Software in the Loop* (SIL) and *Model in the Loop* (MIL) set-ups used to verify if an application's concept fits the requirements at all [35]. As V2X systems are cooperative systems which aim for improving traffic safety and traffic flow, an application must be verified in complex traffic scenarios that leverage cooperation. Otherwise, its impact on traffic may never be studied if the application cannot unfold as intended [19]. This also implies that the application needs to be verified in myriad scenarios to cover most if not all corner cases. For example, a traffic jam avoidance application might be able to reduce some type of traffic jam but cause more serious traffic jams in other cases unintentionally. Such problems may only be discovered before deployment if the application is evaluated in complex and diverse driving scenarios.

Creating appropriate V2X simulations requires proper tools. One frequently used simulation tool is OMNeT++[1], which provides a *Discrete Event Simulation* (DES) framework for model developers. Well-known V2X models for OMNeT++ are Artery or Veins [31, 34]. Where Artery focuses on ETSI ITS, Veins focuses on US *Dedicated Short Range Communication* (DSRC). Artery does not implement the ETSI ITS stack on its own but employs the open-source library Vanetza to provide standard-compliant message formats, packet generation and handling [32] . The two lowest layers of the stack adhere to IEEE 802.11 and matching models of 802.11 network interface cards are provided by the INET[2] framework [29].

As indicated by Sommer et al. before, realistic V2X scenarios are not possible without a bidirectionally coupled traffic simulation [34]. Hence, Artery as well as Veins use the traffic simulator SUMO to incorporate realistic vehicle behavior [24]. SUMO also enables manipulation of vehicles' behavior by third-party tools such as network simulators. For example, when an emergency break assistant based on V2X communication is simulated, emergency breaks will not only be triggered by the network simulation but also performed by the vehicles in the traffic simulation [27].

Aforementioned simulation models can be used to evaluate a wide range of V2X applications and use cases. Also, rapid prototyping and concepts verifications can be achieved by these simulations. For example, a novel V2X application employing parked vehicles as Global Navigation Satelite System (GNSS) base stations has been investigated using the Artery framework [35]. GÃijnther et al. shows an approach for sharing local sensor data with other road traffic participants using V2X communication [19]. They investigated the feasibility of collective perception using different message formats and vehicle densities. These are only two examples how simulation can be used to check if issues exist in application designs even before any HIL test cases are created.

2.3 Conformance and Interoperability Testing

Interoperability and conformance are two of the main concerns when thinking of vehicles communicating with each other. Independent of manufacturer or developer, every vehicle has to "speak the same language" to enable an useful information exchange. ETSI provides extensive guidelines to test implementations for conformance and interoperability.

2.3.1 *Conformance Testing.*

Conformance testing ensures that an implementation conforms to a certain protocol specification [10]. However, it does not check if the specification meets requirements like reliability, performance or robustness. To provide a common test base suitable for every stack implementer, conformance tests themselves must be standardized. Hence, the test execution and test cases are based on the extra information provided by ETSI. For example, the conformance test process (including predefined request forms) for the CA service is described in [12–14]. Furthermore, ETSI includes *Testing and Test Control Notation* (TTCN-3) test specifications which can be used to validate a protocol [14]. TTCN-3 is an abstract testing language, specifically invented for testing communication systems [38]. Among others, it supports timers and decisions to model the expected communication behavior. Its test automation facilitates to check implementations against these expectations.

Even though ETSI provides a plethora of TTCN-3 test cases, conformance tests are not meant to constitute exhaustive tests [10]. Especially for networks with high node mobility it is not possible to provide test cases for every circumstance using non-automated test case generation. Hence, conformance testing can only be seen as first step to check if a tested implementation fits the standards.

2.3.2 *Interoperability Testing.*

Interoperability testing is meant to ensure that ITS stations by various manufacturers can actually communicate with each other. Hence, after ensuring standard conformance of an implementation, interoperability tests are the logical next step. As ETSI leaves some room for configuration, the *CAR 2 CAR Communication Consortium* (C2C-CC) tailors the system by its *Basic System Profile* (BSP) and refines the triggering conditions for *Decentralized Environmental Notification Messages* (DENMs) [5–9]. These additional documents define the behavior more precisely in cases where ETSI ITS specifications set only a framework but not entirely precise actions. For example, ETSI defines how a DENM is structured, but does not specify when a particular DENM should be generated [16].

According to ETSI, the test candidate for ITS conformance and interoperability testing is an *Implementation Under Test* (IUT) [11]. The most obvious way to test such an implementation would be SIL testing. This means, putting an implementation (the ETSI ITS protocol stack) into a test framework. However, an IUT is usually tied to a particular *System Under Test* (SUT) which is then connected to a testing framework instead [11]. Testing a whole V2X system allows the test framework to instrumentalize common interfaces such as *Controller Area Network* (CAN), radio and GNSS. This eases the effort to integrate a test subject as SUTs may employ similar interfaces to interconnect with surrounding devices.

3 V2X HIL SYSTEMS

HIL systems for V2X ECUs are the logical next step of testing in the V2X environment. Surely, SIL or MIL systems are great to evaluate new applications or standards and can be used for rapid prototyping or development. However, if safety-critical systems are

---

[1] https://omnetpp.org/

[2] https://omnetpp.org/download-items/INET.html

to be introduced in vehicles, the ECU and the software running on that ECU must be verified in combination [18].

As already shown in Section 2.1, a HIL test bed depends on arbitrary realistic simulation of sensor input. In current ADASs the perception of a car is quite limited. This means, if only radar or camera sensors are available, the recognizable environment by a car is limited to the field of view of these sensors [33]. In Figure 3 the ego vehicle (depicted in red) is equipped with radar sensors (indicated by the green cones). All vehicles perceived by the ego vehicle are tinted green. Due to limited scope of radar it is sufficient to simulate only the green vehicles to create a realistic environment for the ego vehicle. However, when V2X communication is incorporated as a new sensor type, the range of environmental perception of the ego car increases dramatically. On the analogy of the radar-only set-up, at least all V2X-capable vehicles within the communication range must be simulated in addition to create a realistic environment in which the V2X ECU can be verified.

Figure 4 shows typical hardware interfaces a V2X HIL simulation needs to support for input provision to an embedded DUT. To make V2X applications function properly, various vehicle parameters like velocity, curvature, longitudinal and lateral acceleration, position etc. are required [17]. Therefore, these parameters must be passed to the DUT at a suitable rate using the interfaces designated by the DUT, e.g. CAN, GNSS and a V2X radio interface. Wireless V2X data frames of the simulated environment can be passed on to the DUT via the V2X radio interface. The content and timing of these frames is determined by the software model running the current test case.

What happens with data generated by the DUT depends on the HIL architecture. On the one hand, an open-loop architecture allows to validate the DUT's behavior based on its reactions on the given stimuli without feedback. On the other hand, closed-loop simulation allows for online evaluation of the DUT behavior. All actions the DUT performs are reflected by the simulated environment, e.g. surrounding vehicles may adapt their behavior according to DUT's last actions.

3.1 Closed-Loop Systems

Closed-loop systems are characterized by the fact that they adapt dynamically to the reactions of a DUT. Hence, for a certain test run only basic behavior is pre-defined, e.g. vehicles' destinations are given but the route may change during the run. Likewise, environmental circumstances like road conditions or simulated accidents may be seen as baseline of a test scenario. With this pre-defined data a test run can be initiated. If a test run is executed without an attached DUT, the outcome and runtime behavior such as the exchanged V2X messages should be equal for each run [1]. However, if a certain DUT is added to the test set-up, the message exchange will be different because the simulated vehicles adapt their sending behavior according to the external information by the DUT.

To conduct closed-loop tests, two base requirements must be fulfilled: (1) The software model must represent a sufficiently realistic V2X environment and (2) this model has to be executed in real time. Either of the DES frameworks for VANETs mentioned in Section 2.2 can be used as a quite accurate simulation system.

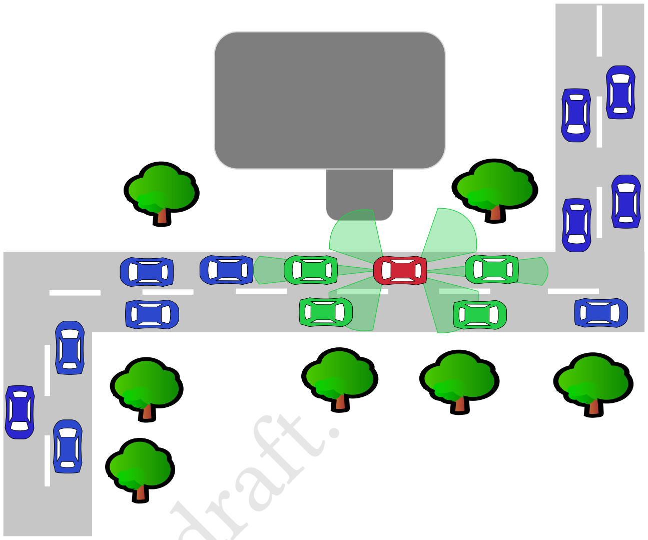


Figure 3: Environment objects detected by the radar sensors (depicted in green) and by V2X communication (all depicted vehicles)

We have shown in earlier work that Artery can be coupled with an external DUT after some modifications [28]. A *Field Programmable Gate Array* (FPGA) powered SDR was used to interconnect the DUT and the simulation environment on the radio channel and GNSS positions were streamed via Ethernet to the DUT. While it has been possible to demonstrate the exchange of messages between test environment and DUT, this approach is limited by the number of surrounding vehicles. On standard computers up to five vehicles can be added, otherwise the (soft) real-time execution fails.

Similarly, it is also possible to couple Veins with a DUT [4]. Buse et al. created an interface which communicates with a HIL test bed for this purpose, which is managing the synchronisation between the different simulator components.

However, in both systems, exact real time execution is one of the biggest problems. Buse et al. mention periods in which the vehicle dynamic simulation has to wait up to four milliseconds until updates can be provided by the synchronisation interface. This is consistent with our own earlier observations [26], where we conclude that DESs are not able to simulate bigger scenarios in real time. Even more, wireless models for DES tend to produce many events at nearly same time instances because one sending event causes many receiving events. This complicates to predict the real-time capabilities of a test run.

Thus, it can be concluded that current closed-loop systems are not capable to simulate arbitrary complex scenarios. This is related to the issue that no proper solution to parallel wireless models for DESs is currently available [25]. Hence, closed-loop systems are not suitable for complex scenarios at present time.

3.2 Open Loop Systems

An open loop HIL is characterized by its fixed environment behavior. Actions and reactions of the DUT during the test run are not reflected by the simulation. Hence, open loop HIL tests rely

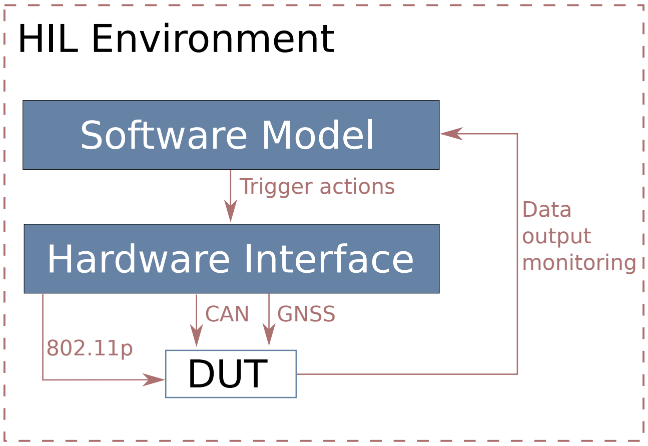


Figure 4: A V2X HIL has to serve the DUT’s interfaces. These typically include CAN, GNSS and IEEE 802.11p radio.

solely on pre-defined environment behavior. How these pre-defined scenarios are created depends on the used toolchain. For example, a test developer could define possible time points or situations in which specific messages should be generated. However, this method of generating test scenarios needs a very deep knowledge about the applications to test, as the test developer has to mimic realistic behavior of the environment vehicles. Alternatively, environment’s behavior can be simulated and recorded by one of aforementioned V2X simulators. To figure out which messages must be forwarded to the DUT during the test run, one vehicle in the simulation is chosen to be the simulated representation of the DUT (further on called physical twin). Each time the physical twin receives a message in the simulated scenario, this message is captured. During the test run, these pre-recorded messages are played back to the DUT. While playing back the messages, the reactions of the DUT can be monitored and evaluated, but the playback itself is not affected.

Despite the fixed playback, open loop testing might still be a suitable approach to test some ADASs. For example, if a radar-based break assistant is to be evaluated, it is sufficient to pre-record the driving behavior of a vehicle in front of the ego vehicle. The tested application is expected to depend only on the distance information produced by its local radar sensor. The reactions of the DUT by no means influence the behavior of the vehicle ahead. In this case, no feedback between both vehicles is required.

However, when speaking of cooperative scenarios powered by V2X communication, not only the behavior of the DUT can be altered during a test run. As V2X communication is mostly broadcast information exchange, it may initiate behavior changes for any networked road participant. In other words, not only the DUT has to adapt its behavior, but also the neighboring vehicles can be influenced by the DUT. Therefore, pre-calculation of a test scenario requires to predict the information stream generated by the DUT als well as the reactions resulting from this information stream. Prediction issues related to vehicle dynamic and information flow are discussed in detail in Section 4.

3.3 Semi-Closed Loop Systems

A semi-closed loop simulation approach can be seen as a intermediate stage between closed-loop and open-loop. It relies on a interactive replay scheme which allows for changing the time point at which certain messages are triggered and therefore sent to the DUT. For example, let us assume a message exchange between a simulated car and the DUT, in which the simulated car replies to a message generated by the DUT. The interactive replay would delay the reply until the message from the DUT has been received beforehand. While the reply is predefined as in open-loop systems, its transmission is not triggered by time but another message.

However, in V2X communication scenarios, the vehicle dynamics can be closely linked to the content of generated messages. For example, each vehicle encodes its own position up to ten times per second in *Cooperative Awareness Messages* (CAMs) [17]. This correlation of vehicle dynamics and the information transmitted to other vehicles may not be ignored in some test cases.

If event triggered messages such as DENMs are going to be disseminated, the problem may even get exacerbated [16]. While a DEN use case such as "vehicle breakdown" can be triggered when a vehicle reaches a specific road section, interaction between vehicles may shift the actual time point when this section is reached by a few seconds and thus cannot be pre-calculated reliably. Even though the correct order of transmitted messages can be maintained by semi-closed loop systems, it becomes necessary to update at least some parts of the pre-recorded messages. Without updating messages during playback, stale event information may be transmitted albeit in a correct logical order. Depending on the test requirements, it may be prohibitive then to adapt vehicle dynamics without updating messages accordingly. Thus, a semi-closed loop system still shares some issues with the open-loop approach.

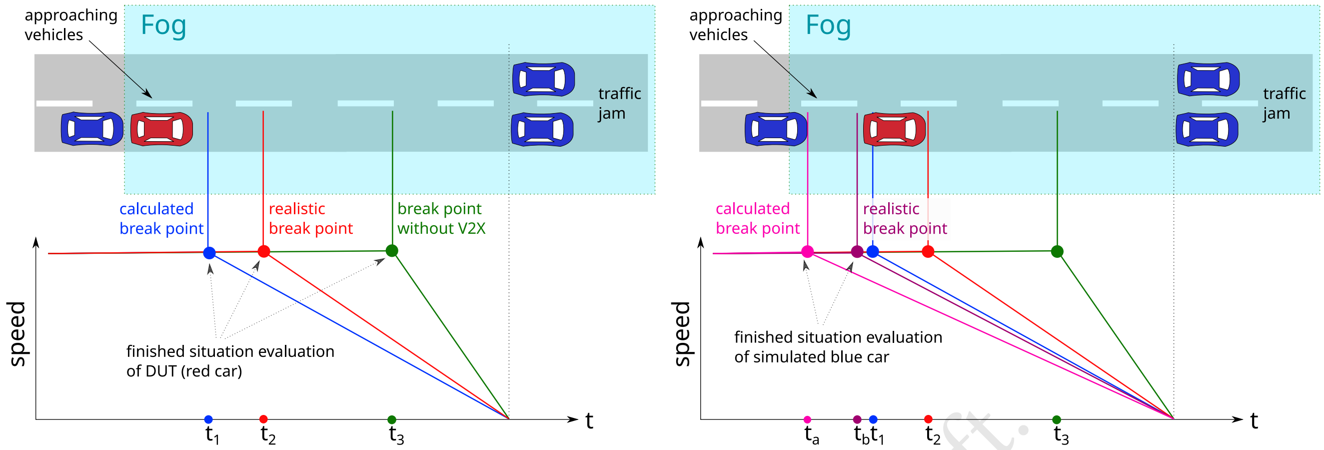
4 DISCUSSION ON ERROR-PRONENESS OF V2X HIL SYSTEMS

As has been outlined in the previous section, none of the three HIL architectures is perfect. If a test designer is aware of the respective system constraints though, he can select the appropriate type for his test requirements. Two particular sources of error with respect to the prediction of vehicle dynamics and the message information flow are discussed in this section.

4.1 Vehicle Dynamics Prediction Error

For clarity, we investigate the driving scenario shown in Figure 5a and 5b to explain the repercussions of wrongly predicted vehicle dynamics: The blue vehicles are purely simulated vehicles which are not existing in the real world. The red vehicle is the simulated representation of the DUT (physical twin). At the right end of the figures, the dangerous end of a traffic jam is shown, respectively. Without V2X communication, this traffic jam can only be detected quite late because the vehicles are covered in heavy fog. A "disconnected" driver would have to perform an emergency break to prevent a rear-end collision solely based on his vision. Hence, the emergency break can only be initiated very late in this case, as indicated by the green vertical bar at t_3 in Figure 5a.

To avoid such a dangerous scenario, the presence of the traffic jam can be disseminated via V2X communications [9]. Traffic jam



(a) Changing dynamics of the DUT

(b) Cascading effects of prediction errors

Figure 5: Changing vehicle dynamics due to prediction failures in open-loop V2X test beds.

information is then received by the physical twin and it can start breaking automatically, shifting the simulated time point of breaking from t_3 to t_1 according to an application model. Thus, vehicle dynamics of all road participants are locked to the expectation that the test subject decelerates at t_1. However, DUT's runtime behavior likely differs from the model due to processing delays or unmodeled details. These difficulties in predicting the runtime behavior of real-time systems have been observed by Wang Lei et al. and Bate and Burns [3, 37]. Because DUT's actual time point of breaking t_2 cannot be calculated a-priori, a variance between t_1 and t_2 is unavoidable.

The effect of such time point variances with respect to vehicle dynamics is clearly visible in Figure 5a: Stopping the ego vehicle before the traffic jam obviously requires more break force at the later time point as indicated by the steeper slope of the red line. At a first glance, it could be assumed that slight differences in breaking are not affecting the test's quality. However, the vehicle dynamics are entirely pre-simulated in an open-loop test case, creating a source for logic errors: A DUT may still be evaluating the situation when the pre-simulated vehicle dynamics already change. This violates the action (decide to break) and reaction (vehicle decelerates) principle.

The idea to solve this problem by adjusting the dynamics of the ego car according to its shifted breaking time point sounds simple, but it is ill-fated: Since no simulation is running in parallel to the open-loop test's playback, no vehicle trajectories can be recalculated spontaneously. Quite the contrary, Figure 5b highlights the complications of adapting only DUT's vehicle dynamics during an open-loop test run. Even if we were able to adjust the trajectory of the physical twin, its surrounding vehicles still act the way as has been simulated in advance. In the shown case, a car following the ego vehicle maintains a constant safety gap to the physical twin. However, the following car would now decelerate unsolicitedly at t_a. Considering the physical twin's changed behavior it should keep up to it until t_b. It turns out that only a full traffic simulation can handle all the interdependencies of vehicle mobility.

Combining an online traffic simulation with pre-recorded communication scenario is also not without problems: V2X communication is closely linked to vehicle dynamics, e.g. by triggering certain messages or at least by affecting data fields of messages. This justifies the prevailing claim that a bi-directional coupling between traffic and network simulation is mandatory to achieve representative V2X scenarios [34]. However, as already explained in Section 3.1, this is limited to small scenarios at the moment.

4.2 Information Flow Errors

Information flow errors describe system inconsistencies occurring when information is distributed in the presence of prediction errors related to pre-simulated scenarios. Figure 6 shows a common V2X scenario demonstrating this issue: The green vehicle is the original source of a DENM, which is to be disseminated to all other depicted vehicles using *Contention-Based Forwarding* (CBF) [16]. As indicated by the transmission ranges (dotted circles), the green vehicle cannot reach all vehicles directly. The red vehicle represents the DUT, hence its behavior is not known a-priori.

CBF is a distributed routing algorithm where each receiver decides whether it shall re-transmit a packet or not. Each receiver starts a timer before re-transmission, whose timeout duration is based on the distance between the initial sender and the respective receiver: The larger the distance, the shorter the timeout. Hence, the timer of the vehicle with the greatest distance expires first and re-transmits the packet before its sibling receivers. When the same packet is received again while its timer is still active, the packet is discarded, the timer stopped and the vehicle refrains from re-transmitting this packet.

Figure 7a shows the CBF compliant sequence of messages when the scenario of Figure 6 is simulated: The green vehicle initiates CBF by transmitting the first message. This packet is received by the red car starting its CBF timer accordingly. After this timer has expired, the message is forwarded again and by then every car has received the message at least once in this scenario. Thus, the information has reached all addressed vehicles.

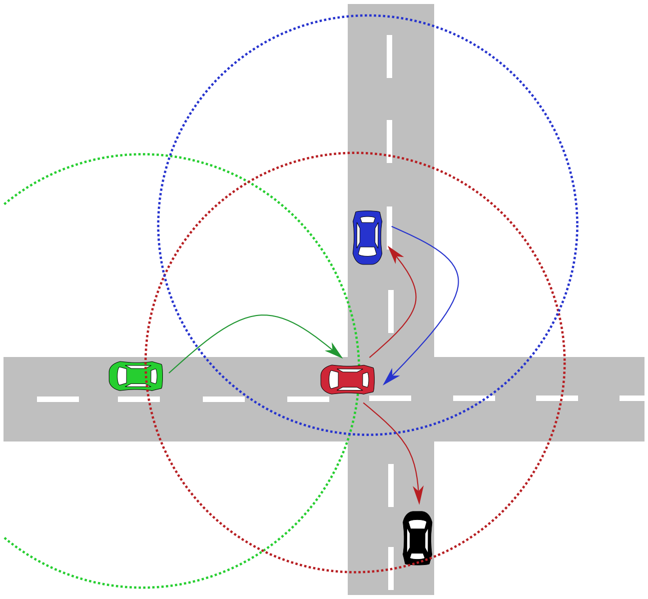
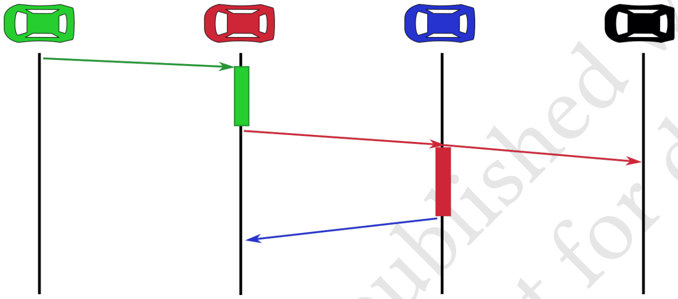
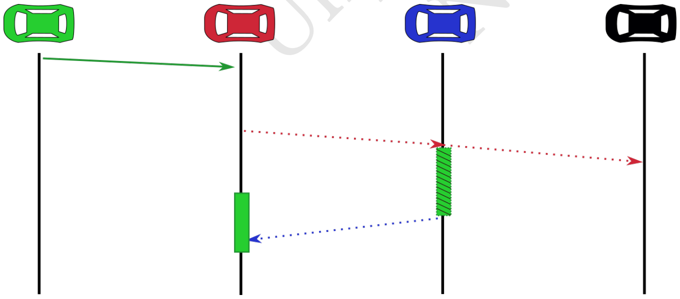


Figure 6: Simple communication scenario. Arrows represent transmissions and correspond to the arrows of same color in Figure 7. Dotted circles feature the transmission range of vehicles in the corresponding colors.



(a) Simulated message flow according to CBF algorithm



(b) Actual message flow with a delayed DUT

Figure 7: Message flow generated by the CBF routing algorithm. Solid arrows represent transmitted messages. Dotted arrows represent transmissions assumed by the environment but are not actually transmitted during test run.

Figure 7b, however, shows a potential outcome in a test run involving a DUT: Again, the green vehicle initiates CBF. Due to internal delays of the DUT as well as delays introduced by the test framework, the DUT (red) does not start its CBF timer immediately. As our test bed is an open loop, this is neither handled nor recognized by the playback environment. Thus, simulated vehicles will still behave in the forwarding process as if the DUT forwarded as expected (red dotted arrows), though DUT’s re-transmission is still outstanding. Consequently, the blue vehicle starts its CBF timer, even though it has not received the message beforehand. However, DUT’s contending timer is now to expire after the blue vehicle’s contention, i.e. DUT’s re-transmission is cancelled when the blue vehicle proceeds with forwarding. The most striking issue is then with the black vehicle, which is not within blue’s transmission range: In a real environment, it would never receive the message at all with described behavior of the DUT. The open-loop test bed ignores this severe logic error and just continues with a black vehicle being aware of information it has actually never received.

The semi-closed-loop approach mitigates such information flow errors occurring in open-loop systems: In the scenario of Figure 7, the semi-closed-loop approach would only proceed with blue vehicle’s forwarding until the prerequisite condition is met, i.e. the to-be-forwarded messages has been received beforehand. Hence, DUT’s contention is not canceled by mistake and the information flow keeps its integrity.

4.3 Probability and Severity Discussion

Besides those two discussed problems V2X HIL test beds are facing, the crucial point is the necessity to execute any model in real time. Without this requirement, closed-loop systems would be the perfect fit for every scenario. However, as current closed-loop models are not capable of handling arbitrarily complex scenarios in real time, test designers have to choose the best fitting architecture for their particular case.

If only a few cars are involved in a test scenario, current closed-loop systems are suitable to evaluate interactive scenarios requiring feedback among vehicles. Unfortunately, closed-loop systems do not scale well as it stands. This leads to the question: How to evaluate more complex scenarios?

Talks with manufacturers and suppliers yielded to our impression, that current focus is still on testing non-safety-critical and less interactive scenarios. Evaluation of these scenarios is possible with an open-loop approach to a certain extent, e.g. if any driver reaction on a V2X event can be neglected. However, one has to keep in mind that some logic constraints, as shown before, will be violated with an open loop, even for day one applications.

Semi-closed-loop systems are a compromise between real-time capability and evaluation depth. For tests focusing merely on the triggering path of V2X messages, this constitutes a suitable approach. However, certain vehicle dynamics errors are unavoidable when vehicle behavior is manipulated due to communication with the DUT. Severity and impact of these errors on the test run is unforeseeable without careful analysis of the specific V2X use case. Runtime measurements with a variety of close-to-production V2X devices and comparing them with the simulated behavior would enable us quantify error margins more accurately. Even with improved error estimation, error-prone test scenarios can hardly be used for evaluation of safety-critical ADASs, though. Hence, fully reactive closed-loop test beds are crucial for reliable test runs in the end.

5 CONCLUSION

This paper has given an overview of current and past research work about HIL simulations. It showed that various test beds share many similarities even if they are used in different areas of application. Every test bed, for example, tries to mimic the real environment as close as possible for the DUT. For similar reasons, an extensive simulation model of the environment is needed in the V2X domain. Such environment models are currently provided as DESs like Artery or Veins. However, these simulation models are too computationally expensive to create a closed-loop V2X test bed. Hence, it was investigated if an open-loop test bed might be sufficient for some V2X application tests.

Flavors of open-loop test beds range from completely open to semi-closed loop, sharing a common feature: The complete test scenario, including the anticipated reactions of the DUT, has to be pre-calculated. As the reactions or actions of the DUT are not entirely predictable, some prediction errors are likely to occur during test. On the one hand, changed message order can cause logic errors where cause and effect of messages is interchanged. On the other hand, erroneously estimated processing times and delays introduced by the test bed may tear apart the timely linkage between vehicle dynamics and V2X messages.

Message order can be preserved by semi-closed-loop systems because they allow to define required message flows. Hence, messages depending on reception of other messages are only triggered if the respective precondition is fulfilled during test. However, vehicle dynamics are often closely linked to V2X messages, so postponing messages can deteriorate the data quality of encoded vehicle parameters. This is caused by the circumstance that vehicle dynamics can hardly be adapted during an open-loop test run.

In summary, some promising approaches targeting at V2X HIL testing exist already. The best solution would be a real-time capable closed-loop simulation, which can be scaled to large scenarios without sacrifying real-time execution. Since current DES models are not up to this, working on a new generation of simulation models should be the main focus in future work.

REFERENCES

[1] Marko Bacic. 2005. On hardware-in-the-loop simulation. In *Proceedings of the 44th IEEE Conference on Decision and Control*. 3194–3198. https://doi.org/10.1109/CDC.2005.1582653

[2] Paul Baracos, Guillaume Murere, C. A. Rabbath, and Wensi Jin. 2001. Enabling PC-based HIL simulation for automotive applications. In *IEMDC 2001. IEEE International Electric Machines and Drives Conference (Cat. No.01EX485)*. 721–729. https://doi.org/10.1109/IEMDC.2001.939394

[3] Iain Bate and Alan Burns. 1998. Investigation of the pessimism in distributed systems timing analysis. In *Proceeding. 10th EUROMICRO Workshop on Real-Time Systems (Cat. No.98EX168)*. 107–114. https://doi.org/10.1109/EMWRTS.1998.685074

[4] Dominik S. Buse, Max Schettler, Nils Kothe, Peter Reinold, Christoph Sommer, and Falko Dressler. 2018. Bridging worlds: Integrating hardware-in-the-loop testing with large-scale VANET simulation. In *2018 14th Annual Conference on Wireless On-demand Network Systems and Services (WONS)*. 33–36. https://doi.org/10.23919/WONS.2018.8311659

[5] CAR 2 CAR Communication Consortium. 2018. Basic System Profile. Release 1.3.0 (Aug. 2018).

[6] CAR 2 CAR Communication Consortium. 2018. Triggering Conditions and Data Quality - Dangerous Situation. Release 1.3.0 (Aug. 2018).

[7] CAR 2 CAR Communication Consortium. 2018. Triggering Conditions and Data Quality - Exchange of IRCs. Release 1.3.0 (Aug. 2018).

[8] CAR 2 CAR Communication Consortium. 2018. Triggering Conditions and Data Quality - Special Vehicle Warning. Release 1.3.0 (Aug. 2018).

[9] CAR 2 CAR Communication Consortium. 2018. Triggering Conditions and Data Quality - Traffic Jam. Release 1.3.0 (Aug. 2018).

[10] European Telecommunications Standards Institute (ETSI). 1995. ETS 300 406 Methods for Testing and Specification (MTS); Protocol and profile conformance testing specifications; Standardization methodology. (4 1995).

[11] European Telecommunications Standards Institute (ETSI). 2011. EG 202 798 Intelligent Transport Systems (ITS); Testing; Framework for conformance and interoperability testing. V1.1.1 (1 2011).

[12] European Telecommunications Standards Institute (ETSI). 2017. TS 102 868-1 Intelligent Transport Systems(ITS); Testing; Conformance test specifications for Cooperative Awareness Basic Service (CA); Part 1 Test requirements and Protocol Implementation Conformance Statement (PICS) pro forma. V1.4.1 (3 2017).

[13] European Telecommunications Standards Institute (ETSI). 2017. TS 102 868-2 Intelligent Transport Systems(ITS); Testing; Conformance test specifications for Cooperative Awareness Basic Service (CA); Part 2 Test Suite Structure and Test Purposes (TSS and TP). V1.4.1 (3 2017).

[14] European Telecommunications Standards Institute (ETSI). 2017. TS 102 868-3 Intelligent Transport Systems(ITS); Testing; Conformance test specifications for Cooperative Awareness Basic Service (CA); Part 3 Abstract Test Suite (ATS) and Protocol Implementation eXtra Information for Testing (PIXIT). V1.4.1 (3 2017).

[15] European Telecommunications Standards Institute (ETSI). 2010. EN 302 665 Intelligent Transport Systems (ITS); Communications Architecture. V1.1.1 (Sept. 2010).

[16] European Telecommunications Standards Institute (ETSI). 2017. EN 302 636-4-1 Intelligent Transport Systems (ITS);Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality. V1.3.1 (Aug. 2017).

[17] European Telecommunications Standards Institute (ETSI). 2019. EN 302 637-2 Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service. V1.4.1 (April 2019).

[18] Georg Schildbach. 2018. On the Application of ISO 26262 in Control Design for Automated Vehicles. In *Workshop on Safe Control of Autonomous Vehicles (SCAV 2018)*. 74–82. https://doi.org/10.4204/EPTCS.269.7

[19] Hendrik-JÃűrn GÃijnther, Raphael Riebl, Lars Wolf, and Christian Facchi. 2016. Collective perception and decentralized congestion control in vehicular ad-hoc networks. In *2016 IEEE Vehicular Networking Conference (VNC)*. 1–8. https://doi.org/10.1109/VNC.2016.7835931

[20] Amirshahram Hematian, James Nguyen, Chao Lu, Wei Yu, and Daniel Ku. 2017. Software Defined Radio Testbed Setup and Experimentation. In *Proceedings of the International Conference on Research in Adaptive and Convergent Systems (RACS '17)*. ACM, New York, NY, USA, 172–177. https://doi.org/10.1145/3129676.3129690 event-place: Krakow, Poland.

[21] Amirshahram Hematian, Wei Yu, David Griffith, and Nada Golmie. 2019. Performance Assessment of Smart Meter Traffic over LTE Network Using SDR Testbed. In *2019 International Conference on Computing, Networking and Communications (ICNC)*. 408–412. https://doi.org/10.1109/ICCNC.2019.8685535

[22] Yanbo Jia and David Cebon. 2016. Field Testing of a Cyclist Collision Avoidance System for Heavy Goods Vehicles. *IEEE Transactions on Vehicular Technology* 65, 6 (June 2016), 4359–4367. https://doi.org/10.1109/TVT.2016.2538801

[23] Sriram Karunagaran, Karuna P. Sahoo, and Masahiro Fujita. 2015. Hardware in loop testing of an insulin pump. In *2015 IEEE International Test Conference (ITC)*. 1–8. https://doi.org/10.1109/TEST.2015.7342416

[24] Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. 2018. Microscopic Traffic Simulation using SUMO, In The 21st IEEE International Conference on Intelligent Transportation Systems. *IEEE Intelligent Transportation Systems Conference (ITSC)*. https://elib.dlr.de/124092/

[25] Stefan Neumeier, Christina Obermaier, and Christian Facchi. 2017. Speeding up OMNeT++ Simulations by Parallel Output-Vector Implementations. In *Proceedings of the 5th GI/ITG KuVS Fachgespräch Inter-Vehicle Communication (FG-IVC 2017)*. 22–25. http://www7.cs.fau.de/fgivc2017

[26] Christina Obermaier and Christian Facchi. 2017. Observations on OMNeT++ Real-Time Behaviour. *CoRR* abs/1709.02207 (2017). arXiv:1709.02207 http://arxiv.org/abs/1709.02207

[27] Christina Obermaier, Raphael Riebl, and Christian Facchi. 2017. Dynamic scenario control for VANET simulations. In *2017 5th IEEE International Conference on*

Models and Technologies for Intelligent Transportation Systems (MT-ITS). 681–686. https://doi.org/10.1109/MTITS.2017.8005599

[28] Christina Obermaier, Raphael Riebl, and Christian Facchi. 2018. Fully Reactive Hardware-in-the-Loop Simulation for VANET Devices. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*. 3755–3760. https://doi.org/10.1109/ITSC.2018.8569663

[29] Institute of Electrical and Electronics Engineers. 2012. IEEE 802.11 Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. (3 2012).

[30] European Technology Platform on Smart Systems Integration. 2015. European Roadmap - Smart Systems for Automated Driving. (4 2015).

[31] Raphael Riebl, Hendrik-JÃűrn GÃijnther, Christian Facchi, and Lars Wolf. 2015. Artery: Extending Veins for VANET applications. In *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*. 450–456. https://doi.org/10.1109/MTITS.2015.7223293

[32] Raphael Riebl, Christina Obermaier, Stefan Neumeier, and Christian Facchi. 2017. Vanetza: Boosting Research on Inter-Vehicle Communication. In *Proceedings of the 5th GI/ITG KuVS Fachgespräch Inter-Vehicle Communication (FG-IVC 2017)*. 37–40.

[33] S. Sivaraman and M. M. Trivedi. 2013. Looking at Vehicles on the Road: A Survey of Vision-Based Vehicle Detection, Tracking, and Behavior Analysis. *IEEE Transactions on Intelligent Transportation Systems* 14, 4 (Dec. 2013), 1773–1795. https://doi.org/10.1109/TITS.2013.2266661

[34] Christoph Sommer, Reinhard German, and Falko Dressler. 2011. Bidirectionally Coupled Network and Road Traffic Simulation for Improved IVC Analysis. *IEEE Transactions on Mobile Computing* 10, 1 (1 2011), 3–15. https://doi.org/10.1109/TMC.2010.133

[35] Thomas Speth, Raphael Riebl, Thomas Brandmeier, Christian Facchi, Ulrich Jumar, and Ali H. Al-Bayatti. 2016. VANET Coverage Analysis for GPS Augmentation Data in Rural Area. *IFAC-PapersOnLine* 49, 30 (2016), 245 – 250. https://doi.org/10.1016/j.ifacol.2016.11.112

[36] Ryan C. Underwood, Bruce M. McMillin, and Mariesa L. Crow. 2008. An Open Framework for Highly Concurrent Real-Time Hardware-in-the-Loop Simulation. In *2008 32nd Annual IEEE International Computer Software and Applications Conference*. 44–51. https://doi.org/10.1109/COMPSAC.2008.165

[37] Wang Lei, Wu Zhaohui, and Zhao Mingde. 2004. Worst-case response time analysis for OSEK/VDX compliant real-time distributed control systems. In *Proceedings of the 28th Annual International Computer Software and Applications Conference, 2004. COMPSAC 2004.* 148–153 vol.1. https://doi.org/10.1109/CMPSAC.2004.1342819

[38] Colin Willcock. 2011. *An Introduction to TTCN-3*. Wiley.

[39] Hao Zhu, Ka-Veng Yuen, Lyudmila Mihaylova, and Henry Leung. 2017. Overview of Environment Perception for Intelligent Vehicles. *IEEE Transactions on Intelligent Transportation Systems* 18, 10 (Oct. 2017), 2584–2601. https://doi.org/10.1109/TITS.2017.2658662