Connected Vehicle Perception Monitoring: A Runtime Verification Approach for Enhanced Autonomous Driving Safety

Redge Melroy Castelino[a], Karina Rothemann[b], Arne Lamm[c] and Axel Hahn[d]
Institute of Systems Engineering for Future Mobility, German Aerospace Center, Escherweg 2, 26121 Oldenburg, Germany

Keywords: Runtime Verification, Collective Perception, Vehicle Environment Perception.

Abstract: Modern autonomous vehicles rely heavily on complex sensor systems for perception tasks, including Advanced Driver Assistance Systems and Autonomous Driving Systems. Accurate sensor perception is essential to ensure the safety of these systems, especially as the level of automation increases. External sensors from the infrastructure or other vehicles can provide useful information to verify the trustworthiness of on-vehicle sensors using V2X communication. This paper presents a novel concept of runtime monitoring to verify the performance of ADS perception systems, taking advantage of the design diversity of connected vehicles and infrastructure based perception sensors in Intelligent Transportation Systems. The proposed approach uses standardised V2X services, such as Collective Perception Service and Location Service from connected participants to estimate a reliable common environment model (CEM) of the driving situation. The established CEM can be used to evaluate the quality of perception of individual road participants during operation, allowing detection and mitigation of system malfunctions of a connected vehicle perception system and enhancing road safety in connected environments. We also discuss open design questions with respect to the perception runtime monitor concept.

1 INTRODUCTION

Modern vehicles are increasingly equipped with active safety systems, including Advanced Driver Assistance Systems (ADAS) which utilize complex sensors to determine the internal state of the vehicle as well as its surrounding environment to support the driver with critical information such as lane departure warning (LDW) or by intervention in vehicle control such as autonomous emergency braking (AEBS). In advanced versions of ADAS, also known as Autonomous Driving Systems (ADS), the responsibility to monitor the driver environment and control the vehicle lies with system. Based upon the distribution of driving tasks between the driver and the system, such as monitoring the driving environment and fallback when the system fails or reaches its limits, the Society of Automotive Engineers (SAE) defines five levels of automation (SAE Standard J3016, 2021).

Nowadays, the safe operation of a vehicle depends increasingly on the fault-free interaction of embedded electronics and software. The resulting requirements to ensure system safety led to the publication of the International Organization for Standardization (ISO) 26262 in 2011. This is an extension of International Electrotechnical Commission (IEC) 61508 for the application of electrical/electronic (E/E) systems in road vehicles, which provides specifications and processes for ensuring functional safety. Similarly, the ISO 21448 standard discusses Safety Of The Intended Function (SOTIF) of ADAS and ADS to ensure exclusion of unreasonable risk due to hazards from functional deficiencies of the intended function.

Ensuring operational safety is especially challenging from SAE level 3 automation onwards, since the driver no longer performs any active driving tasks during operation and it is the system's responsibility to monitor its driving environment and recognise when it reaches its operational limits, so that an appropriate fallback mechanism can take over. This implies rigorous safety validation demands for such systems. However, despite various advancements, the greatest challenges for deployment of ADS at higher levels of automation is the infeasibility of complete exhaustive testing during design time verification (Koopman and

[a] https://orcid.org/0009-0001-2016-3988
[b] https://orcid.org/0009-0000-7030-7962
[c] https://orcid.org/0000-0002-8815-3444
[d] https://orcid.org/0000-0003-2240-5351

Castelino, R., Rothemann, K., Lamm, A. and Hahn, A.
Connected Vehicle Perception Monitoring: A Runtime Verification Approach for Enhanced Autonomous Driving Safety.
DOI: 10.5220/0012696400003702
Paper published under CC license (CC BY-NC-ND 4.0)
In *Proceedings of the 10th International Conference on Vehicle Technology and Intelligent Transport Systems (VEHITS 2024)*, pages 402-409
ISBN: 978-989-758-703-0; ISSN: 2184-495X
Proceedings Copyright © 2024 by SCITEPRESS – Science and Technology Publications, Lda.

Wagner, 2016).

The recent UL4600 standard provides an overview of various safety principles, tools, techniques and life cycle processes to build and evaluate a safety argument of an autonomous vehicle (AV) (UL 4600, 2022). In addition to various verification and validation techniques during design and development such as software unit testing, stress testing, model-in-the-loop (MiL) testing, software-in-the-loop (SiL) testing, Hardware-in-the-loop (HiL) testing, the standard recommends runtime monitoring of safety related operational faults and design assumption violations.

Current approaches of assuring ADS safety includes identification of safety requirements using appropriate Hazard and Risk Analysis (HARA) techniques and verification of safety requirements with rigorous design time verification complemented with runtime verification approaches.

For safety critical automotive systems, even with a well formulated specification and implementation, runtime faults can arise from unanticipated operating conditions, maintenance errors, runtime faults, malicious attacks and other sources (Koopman, 2011). Therefore, to ensure safe fault-free operation of ADS, it is essential to complement design time verification with runtime verification.

Consider ADS perception, a system failure, such as false negative detection of a road user could lead to a life threatening situation during its operation. While ADS perception is subjected to rigorous design time testing, it is infeasible to test in all variations of operating conditions. Typically, such safety critical system requirements that cannot be design-time verified in all operating conditions are subject to runtime verification to assure safety. Runtime monitoring is the detection of anomalies in a target system by comparing the observed state of the system to an expected state of the system (Delgado et al., 2004). However, estimating the expected state of a complex system such ADS perception during its operation is not a trivial task. In this paper, the authors propose a novel approach to detect degradation of environment perception quality of Connected Autonomous Vehicles (CAVs), by estimation of a collective expected state of the system by leveraging developments in Vehicle-to-Everything (V2X) services to compare the environment representations of multiple connected vehicles and infrastructure sensors. We further discuss design challenges for a safe perception monitor design and estimation of the expected state of the environment around the AV using V2X services.

We continue this paper in section 2 with a discussion of works related to runtime verification. In section 3, we introduce our runtime verification concept using standardized V2X services to monitor ADS perception and in section 4, we discuss challenges and open questions to ensure a safe monitor design, which is followed by a conclusion in section 5.

2 RUNTIME VERIFICATION FOR SYSTEM SAFETY

In this section, we briefly examine runtime verification (RV) in the context of software and embedded systems. We further discuss RV in the context of automotive embedded systems and the systems of systems approach to RV adopted in general aviation systems.

RV is a verification technique that complements traditional design time verification approaches of a system such as theorem proving, model checking and testing by monitoring the execution of the system under observation (SuO) to check if it satisfies or violates a specification or correctness property (Leucker and Schallhart, 2009). An example of RV is the simplex architecture, where the monitor verifies whether the executing SuO satisfies a specification and upon detection of an anomaly, triggers a fail safe mechanism, which switches control to a more conservative and trusted component that can steer the SuO to a safe state (Sha, 2001).

By using monitors during system deployment, RV acts as a tool to improve system safety and reliability by acting as a safety net around the monitored application (Nelissen et al., 2015). Concepts of runtime monitoring are applicable for both software systems as well as cyber-physical systems. A common implementation of RV for safety critical software is injecting the monitoring code into the application code of the SuO (Chen and Roşu, 2007; Barringer et al., 2004). When the monitoring code is embedded into target code or system and thus consequently utilizes the resources of the monitored program or system, it is referred to as inline monitors. Conversely, monitoring can be executed as a separate process or thread on a separate machine, also known as offline monitors. The various possible design approaches for a runtime monitor, along with nomenclature are elaborated in detail in (Bartocci et al., 2018; Pike et al., 2011; Delgado et al., 2004). One approach for safety critical embedded systems is the commander/monitor (com/mon) approach, wherein a commander or SuO, monitor and fall back system are implemented as fault containment regions (FCR). When the monitor determines the output of the commander to be unsafe, control is passed on to a fallback system. An application of such a com/mon paradigm for ADS trajectory planning tasks is discussed in (Mehmed et al., 2020).

Based upon the functional decomposition of tasks an ADS performs, one of the first and most common functional paradigms for automated vehicles is the sense-plan-act paradigm (Anderson et al., 2014). Based upon this paradigm, runtime monitors for automotive embedded systems can be categorized into five types (Mehmed, 2020).

Type 1 monitors are associated with classic vehicle embedded systems such as an engine control system (ECS) or anti-lock braking. These runtime monitors are employed to observe relevant system parameters such as engine coolant temperature for the ECS to avoid engine overheating. When the temperature exceed a certain threshold, the monitor instructs the ECS to adopt an appropriate risk mitigating action such as activating the radiator fan or informing the driver via the dashboard. Similarly, monitoring of extra functional properties during runtime via structural contracts increase the reliability of designed extra-functional multi-domain models (Nitsche et al., 2017).

Some monitors adopt prognostic frameworks to estimate the degradation of coupled systems or components to facilitate timely maintenance (Sankavaram et al., 2013). Runtime monitors can also be used to verify compliance to regulations such as vehicle exhaust emissions during operation to detect any fraudulent behaviour by the system under test (SuT) during standardized test conditions to curb the problem of doped software that alters its behaviour under certain conditions (Hermanns et al., 2018).

Type 2 monitors include approaches such as Built-In Self-Tests (BIST), cyclic redundancy codes (CRC), watch dog timers etc. that verify the overall health of the SuO. These monitors are generally used to detect and mitigate various software and hardware faults that can cause a system to perform sub-optimally. In automotive networks, parity check codes, arithmetic checksums and CRC are commonly used for error detection to consequently trigger a re-transmission of the corrupted message (Rahmani et al., 2007). Other examples of *Type 2* monitors include software watchdogs that monitor individual timing constraints of applications and their program flow to detect timing faults and provide information for fault treatment or containment (Chen et al., 2007), BIST to detect communication failures associated with the airbag system of a vehicle using read data by identifier (RDBI) and write data by identifier (WDBI) services of airbag control unit (Sasikumar et al., 2011).

Type 3 runtime monitors verify the performance of ADS perception and detect failures associated with the sense or perception operation of the ADS.

Type 4 runtime monitors verify the output from the path planning module of the ADS. Some examples include a monitor that verifies the trajectory from the trajectory planning modules of the ADS under observation, to ensure that the proposed trajectory does not lead to any collision with road obstacles or drive into a non-drivable area and a driving policy verification module that verifies whether the planned driving path conforms to traffic rules and regulations such as road speed limit etc. (Mehmed et al., 2019).

Type 5 monitors verify the safe operation and health of the actuators. For example, verifying the hydraulic brake pressure is in desired range when brakes are actuated.

In general aviation, a challenge in the introduction of commercial off-the-shelf (COTS) autopilot systems in small aircrafts, that had the potential to dramatically reduce the risk of accidents due to human error, was the high amount of cost and time involved in traditional exhaustive design time verification. In order to allow retro-fitment of COTS to small aircrafts, runtime assurance systems (RTA) were introduced (Hook et al., 2018). Similar to the simplex architecture, RTA systems utilize runtime monitors to verify safety critical system parameters of the SuO during operation and trigger appropriate fail safe mechanisms upon detection of an anomaly or malfunction. RTA systems can be implemented at various levels of system abstraction. For example, RTA can be applied to a component that belongs to a larger system such as a RTA system to monitor the performance of an uncertified turbofan engine controller that estimates engine thrust with an optimal tuned kalman filter (OTKF). Here, a runtime monitor is employed to monitor critical engine parameters and trigger an appropriate fallback mechanism, such as regulation of fuel flow to prevent the engine from entering an unsafe state (Schierman et al., 2018). Another approach involves functional decomposition of the system into subsystems, where every subsystem is monitored. Such a multi-monitor approach was used to support certification of unmanned and autonomous flight systems (Hook et al., 2018). Here multiple runtime monitors, such as ground collision avoidance system and no-fly zone avoidance geofence system monitor ensure safe operation of a complex uncertified waypoint planning module. Every monitor evaluated the safety of aircraft operation independently and informed a higher level module called flight executive upon detection or prediction of a breach of its safety boundary to trigger an appropriate reversionary system to bring the aircraft back to a safe state.

The American Society of Testing and Materials (ASTM) describe approaches to safely bound the behaviour of aircraft systems containing complex functions using RTA with detailed requirements for RTA components, best practices and several example implementations in the ASTM-F3269 standard (ASTM F3269-21, 2021). Figure 1 illustrates a nested RTA approach described in ASTM F-3269, with runtime monitors observing safety critical properties at different levels of abstraction (sub-system level and system level) for the introduction of machine learning based components or modules into an aircraft. This approach allowed the introduction of complex systems while relying on the fail safe RTA system to ensure safety of the larger system.

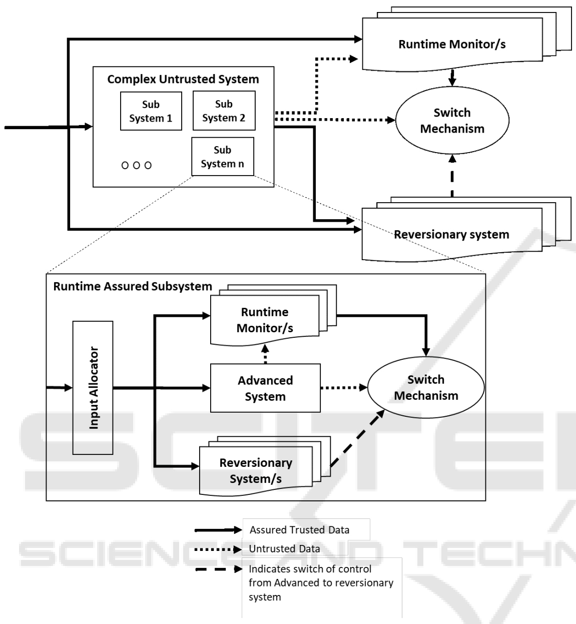

Figure 1: A Nested RTA approach (adapted from ASTM-F3269).

3 A RUNTIME MONITORING APPROACH FOR ADS PERCEPTION

In this section, we present a novel monitoring approach for ADS perception based on standardised V2X services to supplement design time verification and validation.

Typically, autonomous vehicles equipped with V2X communication technology are referred to as CAV (BSI Flex 1890, 2023). In general, CAV's are equipped with On-Board Units (OBU) that facilitate communication with other connected road users. Similarly, vulnerable road users such as pedestrians and bicyclists can be connected to and participate in an ITS environment, for example via ITS applications on hand held devices. Another important participant in the ITS ecosystem is the roadside ITS station, which includes Road Capture Units (RCU) and Road Side Units (RSU). RCUs comprise of infrastructure sensor clusters that observe road traffic. RSUs provide the communication interface between nearby CAVs and RCUs located at roadside (ETSI EN 302 665, 2010). For the purpose of the concept we refer to Collective Perception Service (CPS) (ETSI TS 103 324, 2023) and Location Service (ETSI GS MEC 013, 2022) specified by the European Telecommunications Standards Institute (ETSI). (ETSI TS 103 324, 2023) elaborates the various technical considerations for the realization of the CPS such as the CPS functional architecture, its interfaces in an ITS environment, Collective Perception message (CPM) structures and CPM generation criteria. Similarly, (ETSI GS MEC 013, 2022) facilitates the following functions via the MEC Location Services as follows:

- Location retrieval i.e. location from connected user is retrieved once per request.
- Location subscription i.e. location is retrieved multiple times for each request (event based or periodic).
- Anonymous location report (to collect statistics).
- Location of a certain category of connected users. For example, list of connected pedestrians associated with a specific MEC host.
- List of all connected vehicles that move in or out of a specified location area.

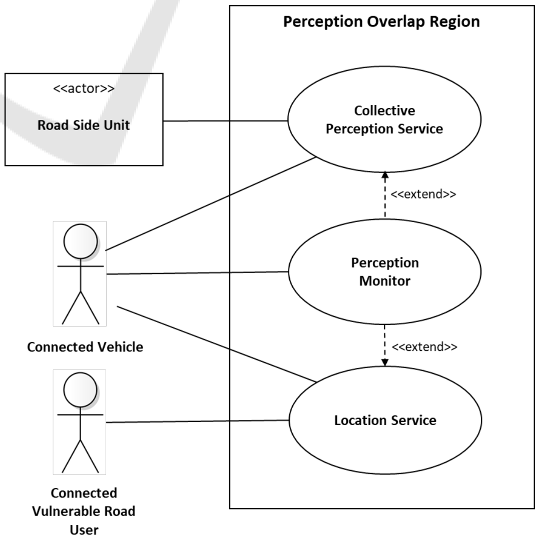

Figure 2: Perception Monitor Use-case diagram.

Figure 2 refers to a use case diagram for the proposed perception monitor. Here, the CAV and RSUs share their perception information, such as objects detected to other connected road participants via the CPS. Similarly, using the location service, CAVs and connected vulnerable road users transmit their location information. The CPS and location services serve as inputs to the perception monitor. Figure 3 illustrates the concept of the ADS perception monitor. Consider a road situation with n CAVs and k RSUs, where there is an intersection or overlap of the perception range of all mentioned connected participants. A road user in this overlapped perception area would be detected by $n+k$ connected participants. The perception information of these individual connected participants would be available via CPS. Furthermore, if the detected road user is a connected participant, its location information would additionally be available via the location service. Using information from these multiple design time verified perception systems, it would be possible to estimate a CEM or common truth of the situation on the road, which can be used as a reference to evaluate the quality of individual CAV perception. For example, false positive detections and false negative or missed detections can be identified during runtime and be used to hand over control to fall back or reversionary systems to assure safety of the CAV.

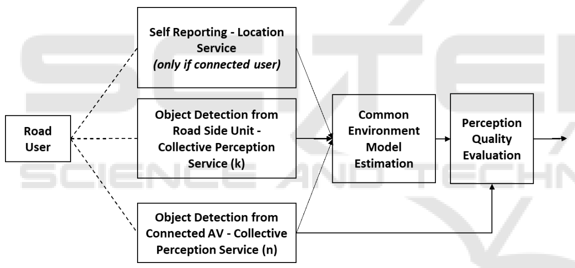
Self Reporting - Location Service
(only if connected user)
Road User
Object Detection from Road Side Unit - Collective Perception Service (k)
Common Environment Model Estimation
Perception Quality Evaluation
Object Detection from Connected AV - Collective Perception Service (n)

Figure 3: Concept of ADS Perception Monitor.

In order to evaluate the described concept, we propose the implementation as described in Figure 4. We first simulate a complex road situation in a high fidelity simulator such as CARLA (Dosovitskiy et al., 2017) or SILAB (Krueger et al., 2005). V2X messages such as CPM and location service messages can then be generated from the perception and positioning data for ever connected participant in the simulation. The data can then be processed to simulate runtime perception faults, such as a false negative detection. Finally, CPM data from all connected participants will be processed via a Perception Monitor module that will first perform the task of estimating a common environment model, which would include the task of data association of objects perceived from various different sources, followed by sensor data fusion of associated objects. The estimated common environment model can then be used to evaluate the quality of CPM from every CAV in the scenario played out to detect any performance deviations of the CAV perception.

4 PERCEPTION RUNTIME SAFETY MONITOR DESIGN ASPECTS

In this section, we discuss various open design aspects and challenges for an ADS perception safety monitor proposed in the previous section as follows:

1. *Independence of a runtime monitor from the safety critical SuO.*
   A typical requirement of RV techniques is the isolation and independence of the runtime monitor from the SuO. A runtime monitor must not change the target system's behaviour functionally, unless the target has violated a specification. Similarly, the RV system must not interfere with the target's timing (Pike et al., 2011). A possible solution to monitor ADS perception is to offload RV functions from the vehicle to infrastructure such as MEC devices or cloud systems. Using V2X services such as CPM and location services, perception information from the vehicle could be transmitted to nearby infrastructure, where the performance of the ADS perception could be monitored while ensuring independence from SuO itself. While an off-vehicle approach can ensure independence of the monitor from the SuO in terms of computational and power resources, it faces challenges to achieve monitoring and fault detection with real time constraints. Alternatively, an on-vehicle runtime monitor design would need to ensure sufficient freedom of interference from the SuO to ensure no consequential impact on the performance of the SuO.
2. *Trade off between observability and monitor independence from the SuO.*
   Observability in the context of runtime monitoring refers to availability of SuO parameters for verification by the monitor (Francalanza et al., 2017). An important consideration for an off-vehicle monitoring approach would be availability of observable system parameters. A runtime monitor implemented on-vehicle could access and observe a wider range of system parameters in comparison to a monitor implemented off-vehicle, which would be constrained to the system parameters observable via available V2X interfaces. On the other hand, in order to ensure safety of the system, it is necessary to ensure independence of the monitor from the SuO in terms of time and space partitioning. An on-vehicle runtime

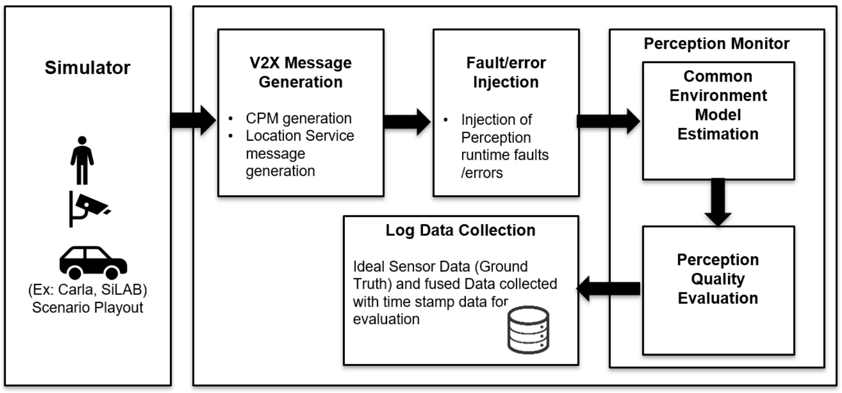
Simulator
(Ex: Carla, SiLAB) Scenario Playout
V2X Message Generation
• CPM generation
• Location Service message generation
Fault/error Injection
• Injection of Perception runtime faults /errors
Perception Monitor
Common Environment Model Estimation
Perception Quality Evaluation
Log Data Collection
Ideal Sensor Data (Ground Truth) and fused Data collected with time stamp data for evaluation

Figure 4: Proposed Implementation.

monitor implementation would have to deal with constraints with respect to bandwidth and computational resources necessary to handle the potentially vast number of V2X messages, especially in a scenario with multiple connected participants involved. A viable solution would be a combination between on-vehicle and off-vehicle runtime monitors, where on-vehicle monitors observe critical parameters or specification of the system for violations or fault detection such as a sensor health check,while an off-vehicle runtime monitors observe performance parameters that are difficult to observe or estimate with just an on-vehicle approach.

3. *Modelling uncertainty of inputs for correctness of monitor.*

   A crucial requirement for a runtime monitor, especially in the context of a safety critical SuO, is that the monitor must ensure a level of correctness themselves (Bartocci et al., 2018). An important consideration to ensure the correctness of a runtime monitor is the quality of input data. For the ADS perception monitor concept described, this means that the correctness of the monitor depends upon the quality of V2X information received. Therefore, it is essential for the runtime monitor to account for the inherent uncertainty of its inputs.

   The CPS as specified in (ETSI TS 103 324, 2023) makes provisions to quantify 3 types of uncertainty in its message data structure: ObjectConfidence (also called existence probability) for every detected object that provide a measure of relevance and quality of the detected object, Class confidence for every detected object that provides a confidence measure in the classification of the detected object and a confidence measure for every attribute in the CPM indicating the state of the detected object, For example: the distance of a detected object is provided with a distance confidence attribute that indicates the accuracy of the distance measured. For the correct functioning of the perception runtime monitor, it is essential that the source of the CPM, identifies and models the these confidence measures to account the various causal factors that contribute to uncertainty or degradation of the source system, such as the presence or absence of adversial operating conditions, measurement accuracy of sensor component etc. An overview of approaches to bound risks stemming from uncertainty in AV perception of its environment and relevant safety metrics to evaluate the accuracy of uncertainty models is discussed in (Benedikt et al., 2024).

4. *Asynchronism and uncertainty due to latency.*

   Estimation of a CEM based on CPM demands the requirement from the CEM estimation module that CPM are processed asynchronously. Although the clocks in the various connected participants may be synchronized with GNSS time, the tasks of sensor data acquisition, processing, message generation and transmission are not synchronous. As a result, the frequency and latency of data arriving from every connected participant may be different. An analysis of the various contributing factors to delay in the collective perception chain including delays due to local perception tasks, communication delays and collective perceptions tasks are discussed in (Volk et al., 2021; Pilz et al., 2023).

   With larger latencies, information received is outdated and requires strategies such as model based prediction to deal with the varying delays from V2X messages, which may consequently introduce uncertainty into the projected or predicted data depending upon the accuracy of the model employed. Consequently, such delays increase the possibility of missed data associations and therefore lead to false positive or false negative monitor outputs. With an asynchronous and offline approach, V2X CPM can be analysed in an offline fashion without having to deal with excessive temporal alignment, thereby potentially improving monitor false detections but deteriorating the real time capability of the monitor. A possible approach for an off-vehicle implementation could include use of dedicated infrastructure sensors to serve as primary inputs to the CEM . Additionally, since the quality of data fusion output deteriorates with the latency of its inputs, it might be beneficial to limit input from vehicle sources based on a threshold latency to improve the accuracy of the CEM. However, such an approach could also lead to exclusion of safety relevant inputs to the CEM, thereby adversely affecting its performance.

5 CONCLUSION AND FUTURE WORK

In this paper, we have discussed various approaches for runtime verification of cyber-physical systems in the context of different domains. Based on this, we were able to introduce a promising approach for runtime monitoring of ADS perception, which uses V2X services such as CPM and location services to continuously evaluate the perception performance of CAVs during their operation. By actively monitoring the system during operation, the proposed approach could potentially detect anomalies, potential failures and deviations from expected behaviour, enabling the system to take corrective actions or warn the driver, thus contributing to overall safety of the ADS operation. Furthermore, a reliable CEM estimation would allow verification of the plan and act phases of ADS, where the planned and executed trajectories could be verified for violations or malfunctions during runtime using the estimated CEM. We propose a first implementation to evaluate the concept using a simulation framework. Our future work would focus on investigating different approaches to model confidence measures for safety critical object information in the CPM data structure while accounting for the various contributing factors to uncertainty of perception information.

ACKNOWLEDGEMENTS

This work was conducted within the Deutsches Zentrum für Luft und Raumfahrt (DLR) internal project 'VMo4Orte - Vernetze Mobilität für lebenswerte Orte', which is funded by the German Federal Ministry for Economic Affairs and Climate Action.

REFERENCES

Anderson, J. M., Kalra, N., Stanley, K. D., Sorensen, P., Samaras, C., and Oluwatola, O. A. (2014). *Autonomous vehicle technology: A guide for policymakers*. Rand Corporation, Santa Monica CA.

ASTM F3269-21 (2021). Practice for methods to safely bound flight behavior of unmanned aircraft systems containing complex functions.

Barringer, H., Goldberg, A., Havelund, K., and Sen, K. (2004). Rule-based runtime verification. In Steffen, B. and Levi, G., editors, *Verification, Model Checking, and Abstract Interpretation*, pages 44–57, Berlin, Heidelberg. Springer Berlin Heidelberg.

Bartocci, E., Falcone, Y., Francalanza, A., and Reger, G. (2018). Introduction to runtime verification. In Bartocci, E. and Falcone, Y., editors, *Lectures on runtime verification*, volume 10457 of *Lecture Notes in Computer Science*, pages 1–33. Springer Berlin Heidelberg, New York NY.

Benedikt, M., Böde, E., Bossert, A., Braband, J., Brade, T., Braun, N., Braun, T., Burton, S., Dallmann, T., Damm, W., Düser, T., Elster, L., Fingscheidt, T., Fistler, M., Franek, M., Fränzle, M., Freyer, J., Galbas, R., Gansch, R., Geyer, D., Haas, L., Haider, A., Heidl, P., Hein, M., Heyl, A., Hiller, J., Hungar, H., Hutter, D., Jung, R., Klein, C., Krüger, J., Kuhn, T., Langner, J., Maurer, M., Mayr, K., Meyer-Vitali, A., Möhlmann, E., Molin, A., Möller, B., Niehaus, J., Nolte, B., Nolte, M., Otten, S., Peleska, J., Peters, S., Poguntke, T., Poprawa, P., Reich, J., Rosenberger, P., Schick, B., Schneider, D., Schneider, S.-A., Schyr, C., Thomas, C., Trapp, M., Wagner, F., Wagener, N., Woopen, T., and Zeh, T. (2024). Controlling Risk for Highly Automated Transportation Systems Operating in Complex Open Environments: A white paper of the SafeTRANS Closing the Gap Initiative.

BSI Flex 1890 (04.2023). Bsi flex 1890 v5.0 connected and automated mobility - vocabulary.

Chen, F. and Roşu, G. (2007). Mop: An efficient and generic runtime verification framework. New York, NY, USA. Association for Computing Machinery.

Chen, X., Feng, J., Hiller, M., and Lauer, V. (2007). Application of software watchdog as a dependability software service for automotive safety relevant systems. In *37th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN'07)*, pages 618–624.

Delgado, N., Gates, A., and Roach, S. (2004). A taxonomy and catalog of runtime software-fault monitoring tools. *IEEE Transactions on Software Engineering*, 30(12):859–872.

Dosovitskiy, A., Ros, G., Codevilla, F., Lopez, A., and Koltun, V. (2017). CARLA: An open urban driving simulator. In *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16.

ETSI EN 302 665 (2010). ETSI EN 302 665 - V1.1.1 Intelligent Transport Systems (ITS); Communications Architecture.

ETSI GS MEC 013 (2022). ETSI GS MEC 013 - V2.2.1 - Multi-access Edge Computing (MEC); Location API.

ETSI TS 103 324 (2023). ETSI TS 103 324 - V2.1.1 - Intelligent Transport System (ITS); Vehicular Communications; Basic Set of Applications; Collective Perception Service; Release 2.

Francalanza, A., Aceto, L., Achilleos, A., Attard, D. P., Cassar, I., Della Monica, D., and Ingólfsdóttir, A. (2017). A foundation for runtime monitoring. volume 10548, pages 8–29.

Hermanns, H., Biewer, S., D'Argenio, P. R., and K\"ohl, M. A. (2018). Verification, testing, and runtime monitoring of automotive exhaust emissions. In Barthe, G., Sutcliffe, G., and Veanes, M., editors, *LPAR-22. 22nd International Conference on Logic for Programming, Artificial Intelligence and Reasoning*, volume 57 of *EPiC Series in Computing*, pages 1–17. EasyChair.

Hook, L. R., Skoog, M., Garland, M., Ryan, W., Sizoo, D., and VanHoudt, J. (2018). Initial considerations of a multi-layered run time assurance approach to enable unpiloted aircraft. In *2018 AIAA Guidance Navigation and Control Conference Kissimmee Florida.*

Koopman, P. (2011). Challenges In Representing CPS Safety.

Koopman, P. and Wagner, M. (2016). Challenges in autonomous vehicle testing and validation. *SAE International Journal of Transportation Safety*, 4(1):15–24.

Krueger, H., Grein, M., Kaußner, A., and Mark, C. (2005). Silab—a task oriented driving simulation.

Leucker, M. and Schallhart, C. (2009). A brief account of runtime verification. *Journal of Logic and Algebraic Programming*, 78(5):293–303.

Mehmed, A. (2020). *Runtime Monitoring for Safe Automated Driving Systems*, volume 324 of *Mälardalen University Press Dissertations*. Mälardalen University, Västerås.

Mehmed, A., Antlanger, M., and Steiner, W. (2020). The monitor as key architecture element for safe self-driving cars. In *2020 50th Annual IEEE-IFIP International Conference on Dependable Systems and Networks-Supplemental Volume (DSN-S)*, pages 9–12.

Mehmed, A., Steiner, W., Antlanger, M., and Punnekkat, S. (2019). System architecture and application-specific verification method for fault-tolerant automated driving systems. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 39–44.

Nelissen, G., Pereira, D., and Pinho, L. M. (2015). A novel run-time monitoring architecture for safe and efficient inline monitoring. In de La Puente, J. A. and Vardanega, T., editors, *Reliable Software Technologies - Ada-Europe 2015*, Programming and Software Engineering, pages 66–82, Cham. Springer International Publishing and Imprint: Springer.

Nitsche, G., Görgen, R., Grüttner, K., and Nebel, W. (2017). Structural contracts – motivating contracts to ensure extra-functional semantics. In Götz, M., Schirner, G., Wehrmeister, M. A., Al Faruque, M. A., and Rettberg, A., editors, *System Level Design from HW/SW to Memory for Embedded Systems*, pages 77–87, Cham. Springer International Publishing.

Pike, L., Niller, S., and Wegmann, N. (2011). Runtime verification for ultra-critical systems. In *Runtime Verification*.

Pilz, C., Sammer, P., Piri, E., Grossschedl, U., Steinbauer-Wagner, G., Kuschnig, L., Steinberger, A., and Schratter, M. (2023). Collective perception: A delay evaluation with a short discussion on channel load. *IEEE Open Journal of Intelligent Transportation Systems*, 4:506–526.

Rahmani, M., Hintermaier, W., Muller-Rathgeber, B., and Steinbach, E. (2007). Error detection capabilities of automotive network technologies and ethernet - a comparative study. In *2007 IEEE Intelligent Vehicles Symposium*, pages 674–679.

SAE Standard J3016 (2021). Taxonomy and definitions for terms related to driving automation systems for on-road motor vehicles.

Sankavaram, C., Kodali, A., and Pattipati, K. (2013). An integrated health management process for automotive cyber-physical systems. In *2013 International Conference on Computing, Networking and Communications (ICNC)*, pages 82–86.

Sasikumar, C., Agrawal, R., Gupta, S., Gupta, S., and Maheshwari, R. (2011). Built in self-test for fault tolerant real time in-vehicle networks through automotive diagnostics. In *2011 International Conference on Emerging Trends in Networks and Computer Communications (ETNCC)*, pages 379–382.

Schierman, J. D., Neal, D., Wong, E., and Chicatelli, A. K. (2018). Runtime assurance protection for advanced turbofan engine control. In *2018 AIAA Guidance Navigation and Control Conference Kissimmee Florida. January 08-12 2018.*

Sha, L. (2001). Using simplicity to control complexity. *IEEE Software*, 18(4):20–28.

UL 4600 (2022). UL 4600: Standard for evaluation of autonomous products.

Volk, G., Delooz, Q., Schiegg, F. A., Von Bernuth, A., Festag, A., and Bringmann, O. (2021). Towards realistic evaluation of collective perception for connected and automated driving. In *2021 IEEE International Intelligent Transportation Systems Conference (ITSC)*, pages 1049–1056.