D-DEG: A Dynamic Cooperation-Based Approach for Reducing Resource Consumption in Autonomous Vehicles

Tobias Kain

Volkswagen AG, Germany. E-mail: tobias.kain@volkswagen.de

Marcel Aguirre Mehlhorn

Volkswagen AG, Germany. E-mail: marcel.aguirre.mehlhorn@volkswagen.de

Hans Tompits

Technische Universität Wien, Austria. E-mail: tompits@kr.tuwien.ac.at

Jullian-Steffen Müller

Volkswagen AG, Germany. E-mail: jullian-steffen.mueller@volkswagen.de

Operating a vehicle autonomously is a resource-intensive task. Since resources, like computing power, energy, and bandwidth, are limited in such vehicles, methods for reducing resource consumption are required. In this paper, we propose D-DEG, a cooperation-based approach for autonomous vehicles that is capable of reducing resource usage. The basis of our approach is that vehicles that are in close proximity and that use the same sensor and software set perceive and compute similar data. The idea is to share information, e.g., sensor data and application outputs, between vehicles using VANET (Vehicular Ad-Hoc Network) technologies. The transferred information is used to achieve resource preservation, whereby our approach aims to reduce resource consumption by degrading sensors and applications. To this end, we introduce the so-called *dynamic-degradation evaluator*. This component analyzes the information received by other vehicles to determine whether sensors and/or applications can be degraded. Besides the data received from other vehicles, the dynamic-degradation evaluator also considers the current operational design domain (ODD) and the system state, which includes, for instance, information about the current resource utilization and the safety level of the vehicle, to determine whether degradation operations can be performed. Those degradation operations can range from decreasing the sampling rate of a sensor or the output rate of applications to shutting down sensors or applications, respectively.

Keywords: autonomous vehicles, VANET technologies, system architecture, resource efficiency.

1. Introduction

Safety and reliability are requirements that many autonomous machines strive to achieve. Especially for autonomous vehicles, i.e., SAE level 4 and 5 vehicles (SAE International, 2018), those requirements are of utmost importance since they influence customer satisfaction and, in return, the success of autonomous driving. Therefore, autonomous vehicles have to implement system architectures that maintain a safe operation. Measures that are applied to increase the safety and reliability of those systems include, for instance, a redundant design of the sensor set or the redundant execution of software applications. However, implementing such measures causes an increasing need for resources such as computing power and energy.

As the available resources, e.g., computing power, transmission bandwidth, and power consumption, in autonomous vehicles are limited, due to, e.g., cost, space, and technical constraints, an economical use of resources is required. Furthermore, reducing resource consumption can also benefit, for instance, the range of electrically operated autonomous vehicles or the lifetime of hardware components. Therefore, methods that focus on reducing the resource consumption of autonomous vehicles are important.

In this paper, we introduce D-DEG, a dynamic cooperation-based degradation approach for autonomous vehicles. The idea of D-DEG is that vehicles driving in close proximity, which perceive the same environment, exchange data using VANET (Vehicular Ad-Hoc Network) technologies. The key component of our method is the so-called *dynamic-degradation evaluator* whose task is to determine whether degradation operations can be performed based on the received information. Furthermore, this component takes also the

Proceedings of the 31st European Safety and Reliability Conference
Edited by Bruno Castanier, Marko Cepin, David Bigaud, and Christophe Berenguer

ISBN: 978-981-18-2016-8; doi:10.3850/978-981-18-2016-8_631-cd

current operational design domain (ODD) as well as the system state of the vehicle into account. Depending on the input information, the dynamic-degradation evaluator can perform degradation actions. In particular, in our D-DEG approach, the evaluator may choose to degrade *sensors* or *applications*. We outline the basic ideas in what follows.

Autonomous vehicles are equipped with multiple sensors, including, for instance, cameras, LiDAR, and radar sensors, in order to observe the surrounding environment. Sensing and processing this information is a resource-intensive task. As stated above, the dynamic-degradation evaluator comprises procedures to degrade sensors. These can be degraded, e.g., if the perceived information by the sensor can be compensated by sensor data received from another vehicle. Based on several factors, including, for instance, the current safety level of the vehicle as well as the driving scenario, different sensor degradation means can be applied. For instance, sensors can be shut down or degraded in their sampling rate. We will illustrate the different levels of degradation in a use case that focuses on the degradation of a radar sensor in different ODDs.

The information perceived by sensors is processed by several software applications that are executed by autonomous vehicles. Multiple functions, including, for instance, perception, planning, and vehicle control services, are necessary to operate a vehicle autonomously. Many of those functions perform highly resource-intensive operations. Furthermore, most of the functions required to operate a vehicle autonomously are safety-critical. Therefore, they are executed redundantly, i.e., the vehicle executes multiple applications that perform the same functionality. Increasing the level of redundancy, in turn, increases the resource requirements.

Concerning the degradation of applications, the dynamic-degradation evaluator may, for instance, degrade an application in case a nearby vehicle executes the same application whereby the output of that application can compensate for the output provided by the degraded application. In general, in D-DEG, applications can be degraded using several degradation actions. For instance, applications can be degraded by reducing their level of redundancy or their output rate. For implementing the latter degradation action, we introduce the so-called *active-low* operation mode.

We will illustrate the different application-degradation types by means of two use cases. In the first scenario, the redundancy level of the given mission-planning function is degraded, whereas the second use case illustrates the degradation of the detection function using the active-low operation mode.

Note that prior to performing sensor and application degradation actions, the impact on the safety level of the vehicle has to be evaluated. Degradation actions that cause a drop of the safety level below an acceptable threshold are not allowed and therefore rejected.

The paper is organized as follows: We start with a discussion on related work in Section 2. Then, in Section 3, we present the overall architecture of our dynamic cooperation-based degradation approach D-DEG for autonomous vehicles. Afterwards, in Section 4, we discuss the method for dynamic sensor degradation in D-DEG, whose aim is to degrade or shut down sensors to save resources and extend the hardware lifespan, while Section 5 presents our method for dynamic application-based degradation for reducing computation and power costs. Section 6 concludes the paper and outlines future work.

2. Related Work

To the best of our knowledge, no approach to dynamically degrade sensors and applications based on context information and data received by nearby vehicles exists. However, since developing energy-efficient vehicles is a well-known challenge (Yuan et al., 2015), some approaches based on information received by other vehicles have been introduced in the past.

For instance, Hexmoor and Yelasani (2018) study resource-efficient platooning approaches. They point out that all vehicles, except for the leading vehicle, can shut down all sensors to save energy. However, a shutdown of all sensors introduces safety risks.

Vahidi and Sciarretta (2018) discuss energy-saving potentials when operating a network of connected vehicles. Their approach aims to decrease fuel consumption by sharing data relevant to the driving condition. The authors predict further energy-saving capabilities in operating vehicles in platoons.

Utilizing the information received by other vehicles is not limited to aim for a reduction of energy consumption. For instance, Kausar et al. (2012) discuss the possibility of sharing the sensor data already processed by a particular vehicle in a way that another vehicle can use this data to process and detect collision courses. This approach causes that vehicles have to process additional data. Consequently, the resource needs of the vehicles increase.

Saxena et al. (2019) introduce an approach where vehicles share the obtained surrounding information with nearby vehicles. The shared information is utilized to validate the obtained environment data of the vehicle. Therefore, each autonomous vehicle represents the perceived environment in a size-efficient data format, which is shared with other vehicles. Other vehicles can fuse the information received by other vehicles into their environment representation.

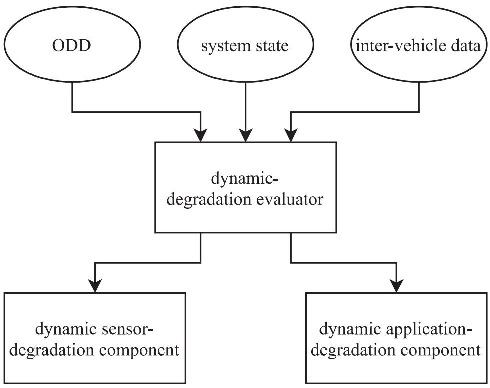

Fig. 1. Overview of D-DEG, the dynamic cooperation-based degradation approach of sensors and applications.

Jia et al. (2016) present an overview of platoon-based vehicular cyber-physical systems. Vehicles with common interests can cooperatively form a platoon. To build and operate platoons, vehicles exchange information using VANET technologies. According to Jia et al., platoons can improve road capacity, safety, and energy efficiency.

Aoki et al. (2020) present a deep reinforcement learning approach for cooperative perception to increase detection accuracy, which aims to reduce the amount of data transferred between vehicles. According to Aoki et al., a cooperative perception can increase road safety as, for instance, vehicles can eliminate blind spots by using the perception data received from surrounding vehicles. A similar approach of a cooperative perception approach using VANET technologies is discussed by Günther (2017).

Note that none of these approaches consider using the data received by other vehicles as well as ODD and system-state information to decide whether sensors and applications can be degraded.

3. Overall Architecture of D-DEG

Autonomous vehicles are equipped with means to perceive the current context which comprises the *operational design domain* (ODD) and the system state. The ODD includes, e.g., information about the road, the traffic infrastructure, temporary manipulations of the road and the infrastructure, static and dynamic surrounding objects, the environment, and information about the availability of digital services (PEGASUS, 2019). ODD information can be, for instance, extracted from sensor data or from data provided by backend services. On the other hand, information from inner-vehicle monitors, e.g., function monitors or computing-node monitors, can be used to infer the system state, which includes, for instance, information about the the current resource utilization, the safety level of the system, or the redundancy level of autonomous driving functions.

Furthermore, autonomous vehicles incorporate VANET (Vehicular Ad-Hoc Network) technologies, like 5G-NR, Wi-Fi, and UWB (Ahangar et al., 2021). These communication technologies enable vehicles to exchange information with other vehicles, the infrastructure, and backend services. For instance, it is conceivable that vehicles exchange their environment model, sensor data, or the output of software applications.

The idea of D-DEG, our approach of a dynamic cooperation-based degradation of sensors and applications, is to analyze the ODD and the system state of the vehicle, as well as the data received by other vehicles, and perform degradation operations based on the provided information. As illustrated in Figure 1, the decision whether a degradation is feasible is made by the so-called *dynamic-degradation evaluator*. Based on the ODD and the system state, as well as the data received by other vehicles, the dynamic-degradation evaluator decides whether a sensor degradation and/or a degradation of the software applications can be performed.

The respective degradations are then performed by the *dynamic sensor-degradation component* and the *dynamic application-degradation component*. Before degradation operations are performed, the dynamic sensor-degradation component and the dynamic application-degradation component have to ensure that planned degradation procedures maintain the safety level of the system, i.e., do not cause a safety violation.

The following sections explain the dynamic sensor-degradation and the dynamic application-degradation in more detail.

4. Dynamic Sensor-Degradation Approach in D-DEG

The idea of the dynamic sensor-degradation approach is to determine whether the current ODD and system state, as well as the data received by other vehicles, allow a degradation of the sensors. Depending on the input data, the dynamic-degradation evaluator can determine different sensor degradation levels. For instance, in case the vehicle is stuck in a traffic jam, some sensors can be conceivably shut down. On the other hand, if the vehicle is, for instance, driving on a highway and is surrounded by vehicles, which can provide data perceived by their sensors, the dynamic sensor-degradation component can potentially instruct some sensors to reduce their sampling rate.

In what follows, we apply the dynamic sensor degradation approach in an experimental setup. Therefore, we define a scenario comprising three ODDs to illustrate the different levels of sensor degradation. Furthermore, we estimate the resource reduction capabilities in a realistic driving

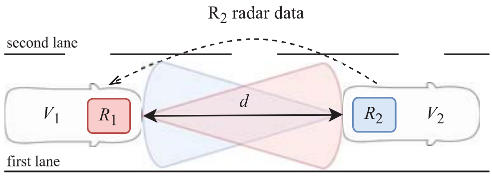

Fig. 2. Scenario of two vehicles driving behind one another.

scenario.

4.1. *Use Case: Radar Degradation*

For this use case, we use the open-source simulator CARLA (Dosovitskiy et al., 2017), which was specifically designed for autonomous driving research, to set up real-world driving scenarios. The main features of CARLA are the variety of open digital assets, the flexible specification of sensor suites, environmental conditions, and full control of static and dynamic actors.

To illustrate the proposed concepts for a dynamic sensor degradation, we designed the scenario shown in Figure 2. The scenario includes two vehicles, referred to as V_1 and V_2. Vehicle V_1 follows for the entire time of the simulation V_2, whereby the distance d between the vehicles depends on the ODD.

Overall, we consider the following three ODDs in this scenario: traffic-calmed areas, urban environments, and highways. In traffic-calmed areas, we assume an approximate distance d between the two vehicles of 15 meters. Furthermore, we assume that d is approximately 25 meters in urban environments, and 50 meters on highways, respectively.

Vehicle V_1 is equipped with a front radar R_1, which scans the area in front of V_1. Furthermore, vehicle V_2 is equipped with a rear radar R_2, which scans the back area of the vehicle. The two radar sensors, i.e., R_1 and R_2, are identical in construction and therefore provide the measured data in the same output format. For the sake of simplicity, we do not consider any other sensors of autonomous vehicles in this use case.

Besides perceiving information using sensors, CARLA also allows extracting data from the simulation itself, such as the speed of vehicles or their location. In a real-world driving scenario, this information can be shared using V2V ("vehicle-to-vehicle") communication. We, therefore, simulate V2V communication by using the simulation data provided by CARLA. Note that the simulated V2V communication does not comprise any latency times.

The vehicles V_1 and V_2 drive with the context-dependent distance d in the CARLA simulation. The rear vehicle V_1 can determine the distance d by evaluating the data measured by radar sensor

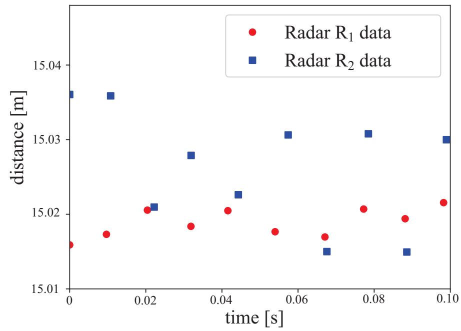

Fig. 3. Sample of the radar data in a traffic-calmed area. R_1 is operating at maximum sampling rate, i.e., R_1 is not degraded. Data provided by R_2 can be used for verification purposes.

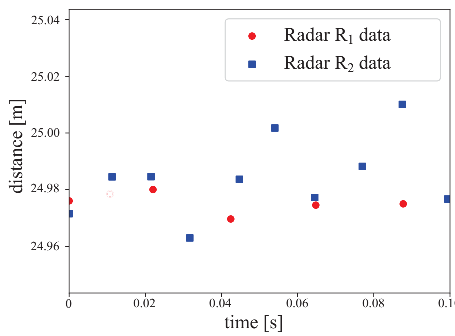

Fig. 4. Sample of the radar data in an urban area. The sampling rate of R_1 is reduced by 50 percent. Data provided by R_2 compensates for the missing data points and can be used for verification purposes.

R_1 or via the V2V communication with vehicle V_2, which shares the sensor information of radar R_2 with V_1. Likewise, V_1 can also share the data perceived by R_1 with V_2.

In the defined scenario, the dynamic-degradation evaluator instructs the dynamic sensor-degradation component to degrade R_1 based on the ODD. Radar sensor R_2, on the other hand, is not degraded. Note that degrading R_2 instead of R_1 would be conceivable as well. In situations in which degradation operation can be performed either by one vehicle or the other, the dynamic-degradation components of both vehicles have to negotiate which vehicle shall apply the degradation measures. Decision criteria that can be considered are, for instance, the battery level of the vehicles, the remaining distance to their final destination, or the total operation time of the vehicles and their sensors. For the sake of simplicity, we

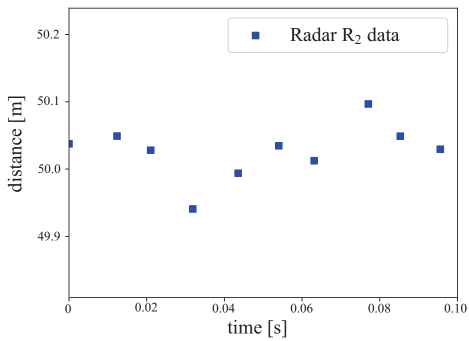

Fig. 5. Sample of the radar data on a highway. R_1 is shut down. Data provided by R_2 compensates the shut down of R_1.

statically define that radar R_1 shall be degraded in this use case.

The degradation of R_1 in the different ODDs is illustrated in Figures 3, 4, and 5. For each ODD, we extracted from the CARLA simulation the measured distance d of R_1 and the received information of R_2 for a 100ms time frame.

We configured the dynamic-degradation evaluator such that in a traffic-calmed area, the sampling rate of R_1 is not degraded as in such an environment it is likely that objects, e.g., children, suddenly appear in front of the vehicles. Consequently, degrading R_1 would cause a safety violation.

The data perceived by R_2, which is received via the V2V communication, can be used for validation purposes. For instance, Wesche et al. (2021) introduced an approach that monitors and validates system components using information from homogeneous and heterogeneous components. In an experimental setup, they illustrate the use of velocity information received from other vehicles via V2V communication to verify the information perceived by a radar sensor heterogeneously.

As can be seen in Figure 3, the distance d between V_1 and V_2 is approximately 15 meters in the considered time frame. The perceived information of R_1 and the received radar data of R_2 from V_2 are in close proximity.

In urban environments, the dynamic-degradation evaluator instructs the dynamic sensor-degradation component to reduce the sampling rate of R_1 to 50 percent. This degradation is justified as in urban environments, the probability of suddenly appearing objects in front of the vehicle is less than in traffic-calmed areas. As can be seen in Figure 4, the radar data of R_2 received from V_2 compensates for the reduced sampling rate of R_1. The data points perceived by R_1 can be used to validate the radar data of R_2 received from V_2.

If the two vehicles are driving on a highway, the dynamic sensor-degradation component of V_1 instructs R_1 to shut down. In this case, as illustrated in Figure 5, V_1 relies on the radar data received by V_2. However, vehicle V_1 can periodically validate the received radar by using the information provided by other sensors, e.g., LiDAR.

Table 1. The cumulated covered distance and traveling time of the vehicles in the different external contexts.

external context	**distance**	**time**
traffic-calmed area	0.5 km	2 min
urban area	4.3 km	10 min
highway	29.2 km	18 min

4.2. *Evaluation of the Resource Reduction*

To evaluate the potential resource reduction by degrading the radar sensor R_1 as described before, we analyze the route between the two German cities Wolfsburg and Brunswick. We use Openrouteservice (Neis and Zipf, 2008), which is an extension of OpenStreetMap (Ramm and Topf, 2010), to identify the covered distances, as well as the traveling time of the vehicles in the three introduced external contexts. Table 1 lists the distance as well as the traveling time of V_1 and V_2.

In total, the journey time from Wolfsburg to Brunswick is 30 minutes. During the entire period, the vehicles are driving on the highway and R_1 is shut down. Furthermore, while driving in an urban environment, the sampling frequency of R_1 is reduced to 50 percent. Consequently, R_1 is operating at the maximum sampling rate only for 2 minutes of the 30 minutes journey, i.e., during the period the vehicle is driving in a traffic-calmed area. Therefore, the overall sampling information of R_1 is reduced by approximately 77 percent.

Assuming that reducing the sampling rate to 50 percent of R_1 also reduces the power consumption of the sensor by 50 percent, the overall power consumption of R_1 decreases by approximately 77 percent. Besides reducing the amount of sampling data and power consumption, degrading R_1 can also increase the lifetime of the sensor. If we assume that operating R_1 with a 50 percent reduced sampling rate doubles the maximum lifetime of the sensor, the lifetime of R_1 is reduced only by seven instead of 30 minutes.

5. Dynamic Degradation of Applications

Besides the dynamic degradation of sensors, our D-DEG approach also aims to dynamically reduce the resource requirements of software applications that are executed by the autonomous vehicle.

5.1. *Basic Approach*

The idea of our method of a dynamic application-degradation in D-DEG is to utilize the data received from other vehicles to enable a degradation of the software applications. Degrading software applications is desirable since it can reduce resource utilization.

Software applications can be degraded on different levels. For instance, a degradation can be achieved by reducing the number of redundant applications. The redundant applications can either be *homogeneous*, i.e., a specific application is executed multiple times, or *heterogeneous*, i.e., multiple different applications offer the same functionality. The redundancy model that we apply defines that applications can be executed in two different operation modes (Kain et al., 2020): *active* and *active-hot*.

Multiple applications can execute the same function, whereby only one application is executed in the active operation mode. An application executed in the active operation mode receives input information from sensors and other functions and can provide output information to functions or control actuators. The redundant applications of a function are executed in the active-hot operation mode. Those applications receive the same input information as the ones executed in active operation mode. However, they do not provide their output to other functions nor control actuators. The output of active-hot applications is only used to validate the output of active applications. Furthermore, active-hot applications can be executed in a degraded mode which requires a lower resource allocation. In case a malfunction of the active application is detected, the operation mode of one of the active-hot applications is upgraded to active.

If the information received by another vehicle can compensate for the output information of active-hot applications, the dynamic application-degradation component can reduce the level of redundancy. Therefore, the dynamic application-degradation component instructs the respective computing node, which executes the active-hot application to terminate that application. Consequently, resources are released.

Besides reducing the level of redundancy, the dynamic application degradation component can also degrade active applications if certain circumstances are met. Therefore, we introduce an additional operation mode called *active-low*. An active-low application has the same characteristics as an application executed in active operation mode. However, active-low applications process the input data with a lower allocation of resources, e.g., a reduced memory or CPU utilization, at the cost of a reduced output rate.

In what follows, we discuss examples that illustrate the reduction of redundant applications as well as the active-low operation mode. Furthermore, we analyze the resource utilization using a realistic software stack.

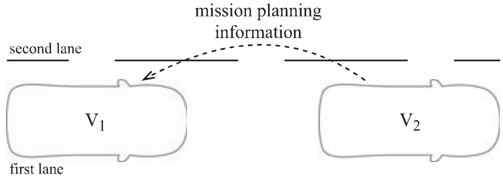

Fig. 6. Scenario of two cars driving behind one another on a highway.

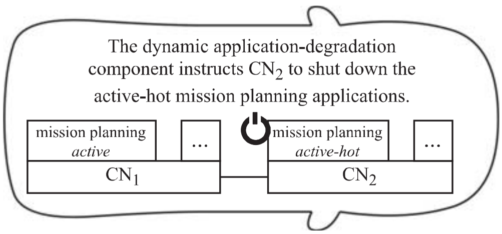

Fig. 7. The configuration of vehicle V_1.

5.2. *Reduction of the Redundancy Level Use-Case*

Assume a scenario, as illustrated in Figure 6, in which vehicle V_1 is following vehicle V_2 on a highway. Furthermore, assume that vehicle V_1 redundantly executes a mission-planning function. As shown in Figure 7, computing node CN_1 runs the active mission planner and CN_2 executes an active-hot mission planner.

As V_1 follows V_2, the mission planning of both vehicles is very similar. Consequently, V_2 can transmit the mission planner output information to V_1 that, in return, uses the received data to validate the output of the active mission planner. As long as the data exchange between V_1 and V_2 is uninterrupted and the internal and external context remain unchanged, the dynamic application-degradation component can shut down the active-hot mission planner.

To analyze the potential resource reduction that can be achieved by degrading the mission-planning application in the presented scenario, we simulated an autonomous vehicle in CARLA using Autoware (Kato et al., 2018), an "all-in-one" open-source framework for autonomous vehicles, which is built on top of ROS 2 Maruyama et al. (2016). Figure 8 shows the general architecture of Autoware.

The mission-planning function is part of the Autoware planning component and includes several ROS 2 nodes. Table 2 lists all the ROS 2 nodes

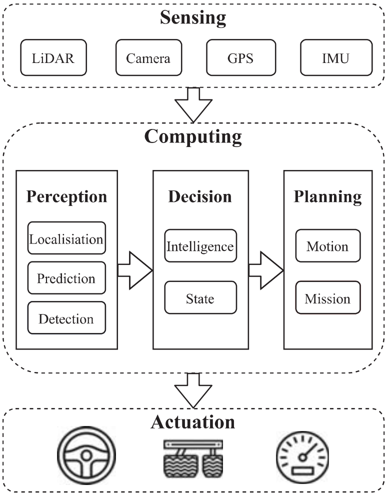

Fig. 8. A general overview of the Autoware architecture.

Table 2. The CPU and memory reassure consumption of the ROS 2 nodes which are responsible for the mission planning.

nodes	CPU	memory
op_global_planer	3.7 %	32 MB
lane_select	2 %	16 MB
lane_rule	1.3 %	35 MB
lane_stop	1.3 %	21 MB

and their CPU and memory utilization which are responsible for the mission planning. Note that we executed the simulation on a computer running Ubuntu 20.04, which is equipped with an Intel® Core^{TM} i7-8850H CPU, an NVIDIA Quadro P2000 Mobile GPU, and 16 GB of memory.

In total, the mission planner consumes approximately 8% of the overall CPU power and about 100 MB of memory. Consequently, by shutting down the active-hot mission planner those resources are released.

5.3. *Degradation of an Active Application Use-Case*

Another scenario we consider is that of three vehicles driving beside one another, as illustrated in Figure 9.

We assume that all three vehicles redundantly execute a detection function. As vehicle V_2 is surrounded by vehicle V_1 and V_3, V_2 will only detect objects that are also detected by V_1 and V_3. Therefore, V_2 can use the detection information received by V_1 and V_3 to incorporate that information in the environment model. Consequently, the active detection application of V_2 can be downgraded to active-low. For instance, this application can lower the frequency at which objects are detected. The output of the active-low application can be used to validate the information received by V_1 and V_3.

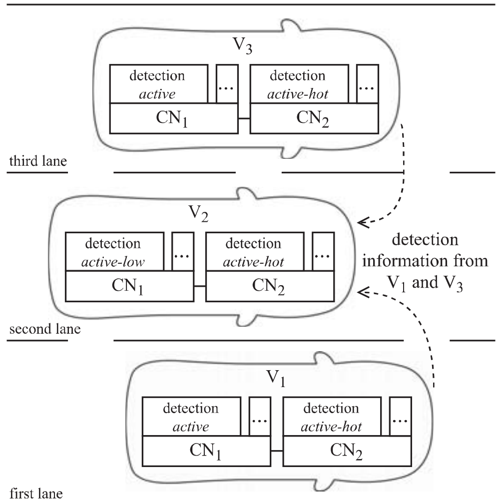

Fig. 9. Scenario of three vehicles driving next to each other.

Table 3. The CPU and memory reassure consumption of the ROS 2 nodes which are responsible for the detection.

nodes	CPU	memory
naive_motion_predict	2.3 %	32 MB
costmap_generator	9.2 %	64 MB
object_roi_filter_clustering	0.7 %	16 MB
lidar_euclidean_cluster_detect	28.8 %	192 MB

Table 3 illustrates the CPU and memory utilization of the ROS 2 nodes, which are part of the detection function. Overall they consume more than 40% of the CPU resources and about 300 MB of memory. Due to this high resource utilization, the detection function offers a high potential to implement means that allow reducing the resource usage.

6. Conclusion

In this paper, we introduced the cooperation-based approach D-DEG for autonomous vehicles for reducing resource usage. The idea of D-DEG is that vehicles that are in close proximity share information that can be used to degrade sensors and applications, whereby we distinguish between different levels of degradations. Sensors, for instance, can be shut down or degraded in their sampling rate. Applications, on the other hand, can be degraded by reducing their level of redundancy or their output rate.

Concerning future work, we plan to integrate D-DEG into our framework of a context-based system architecture (Kain et al., 2020). This framework defines three interconnected layers, which are differentiated by their level of awareness. Furthermore, we plan to test the impacts of our approach on the safety, reliability, and availability of an autonomous vehicle using a simulation environment (Horeis et al., 2020). Besides evaluating the impact of D-DEG on parameters like safety, reliability, and availability, we plan to analyze the concrete resources-saving potential of D-DEG in a case study. One aim of this case study is, for instance, to measure the impact of reducing a sensor's sampling rate on its lifetime.

References

Ahangar, M. N., Q. Z. Ahmed, F. A. Khan, and M. Hafeez (2021). A Survey of Autonomous Vehicles: Enabling Communication Technologies and Challenges. *Sensors 21*(3), 1–33.

Aoki, S., T. Higuchi, and O. Altintas (2020). Cooperative Perception with Deep Reinforcement Learning for Connected Vehicles. In *Proceedings of the 2020 IEEE Intelligent Vehicles Symposium* (*IV 2020*), pp. 328–334.

Dosovitskiy, A., G. Ros, F. Codevilla, A. Lopez, and V. Koltun (2017). CARLA: An Open Urban Driving Simulator. In *Proceedings of the 1st Annual Conference on Robot Learning* (*CoRL 2017*), pp. 1–16.

Günther, H.-J. (2017). *Collective Perception in Vehicular Ad-hoc Networks*. Dissertation, Technische Universität Carolo-Wilhelmina zu Braunschweig.

Hexmoor, H. and K. Yelasani (2018). Economized Sensor Data Processing with Vehicle Platooning. *International Journal of Computer and Information Engineering 12*(6), 428–433.

Horeis, T. F., T. Kain, J.-S. Müller, F. Plinke, J. Heinrich, M. Wesche, and H. Decke (2020). A Reliability Engineering Based Approach to Model Complex and Dynamic Autonomous Systems. In *Proceedings of the 3rd International Conference on Connected and Autonomous Driving* (*MetroCAD 2020*).

Jia, D., K. Lu, J. Wang, X. Zhang, and X. Shen (2016). A Survey on Platoon-Based Vehicular Cyber-Physical Systems. *IEEE Communications Surveys Tutorials 18*(1), 263–284.

Kain, T., J.-S. Müller, P. Mundhenk, H. Tompits, M. Wesche, and H. Decke (2020). Towards a Reliable and Context-Based System Architecture for Autonomous Vehicles. In *Proceedings of the 2nd International Workshop on Autonomous Systems Design* (*ASD 2020*).

Kain, T., H. Tompits, J.-S. Müller, P. Mundhenk, M. Wesche, and H. Decke (2020). FDIRO: A General Approach for a Fail-Operational System Design. In *Proceedings of the 30th European Safety and Reliability Conference* (*ESREL 2020*).

Kato, S., S. Tokunaga, Y. Maruyama, S. Maeda, M. Hirabayashi, Y. Kitsukawa, A. Monrroy, T. Ando, Y. Fujii, and T. Azumi (2018). Autoware on Board: Enabling Autonomous Vehicles with Embedded Systems. In *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems* (*ICCPS 2018*), pp. 287–296.

Kausar, T., P. Gupta, D. Arora, and R. Kumar (2012). A VANET based Cooperative Collision Avoidance System for a 4-Lane Highway. *Notes on Engineering Research and Development, IIT Kanpur Technical Journal 4*, 25.

Maruyama, Y., S. Kato, and T. Azumi (2016). Exploring the Performance of ROS2. In *Proceedings of the 13th International Conference on Embedded Software* (*EMSOFT 2016*).

Neis, P. and A. Zipf (2008). Zur Kopplung von OpenSource, OpenLS und OpenStreetMaps in OpenRouteService.org. In *Angewandte Geoinformatik 2008: Beiträge zum 20. AGIT-Symposium Salzburg*. Wichmann.

PEGASUS (2019). PEGASUS Method: An Overview. `www.pegasusprojekt.de`.

Ramm, F. and J. Topf (2010). *OpenStreetMap–Die freie Weltkarte nutzen und mitgestalten*. Lehmanns Media.

SAE International (2018). Taxonomy and Definitions for Terms Related to Driving Automation Systems for On-Road Motor Vehicles. In *SAE Standard J3016*, pp. 1–16.

Saxena, S., I. K. Isukapati, S. F. Smith, and J. M. Dolan (2019). Multiagent Sensor Fusion for Connected Autonomous Vehicles to Enhance Navigation Safety. In *Proceedings of the 22nd IEEE Intelligent Transportation Systems Conference* (*ITSC 2019*), pp. 2490–2495.

Vahidi, A. and A. Sciarretta (2018). Energy Saving Potentials of Connected and Automated Vehicles. *Transportation Research Part C: Emerging Technologies 95*, 822–843.

Wesche, M., T. Kain, M. A. Mehlhorn, H. Tompits, H. Michalik, A. Dörflinger, and J.-S. Müller (2021). A Context-Based Fault-Detection Approach for Autonomous Vehicles. *Submitted*.

Yuan, X., X. Liu, and J. Zuo (2015). The Development of New Energy Vehicles for a Sustainable Future: A Review. *Renewable and Sustainable Energy Reviews 42*, 298–305.