IEEE Open Journal of
Intelligent Transportation
Systems

Received 18 November 2022; revised 3 March 2023, 25 May 2023, and 23 June 2023; accepted 15 July 2023. Date of publication 1 August 2023; date of current version 8 August 2023.

Digital Object Identifier 10.1109/OJITS.2023.3296812

Collective Perception: A Delay Evaluation With a Short Discussion on Channel Load

CHRISTOPH PILZ[1,2], PETER SAMMER[2], ESA PIRI[3], UDO GROSSSCHEDL[2], GERALD STEINBAUER-WAGNER[1], LUKAS KUSCHNIG[2], ALINA STEINBERGER[1,2], AND MARKUS SCHRATTER[2]

[1]Institute for Software Technology, Graz University of Technology, 8010 Graz, Austria
[2]Department of Electrics, Electronics, and Software, Virtual Vehicle Research GmbH, 8010 Graz, Austria
[3]Kaitotek Oy, 90460 Oulu, Finland

CORRESPONDING AUTHOR: C. PILZ (e-mail: christoph.pilz@v2c2.at)

ABSTRACT Automated vehicles and vehicle-to-everything (V2X) communication open the window for sharing of sensor data. This paper aims to provide a systematic view of the delay chain involved. We implemented collective perception (CP) into two street legal automated driving demonstrators (ADDs) to provide insight into the components' delay. The implementation allowed us to gather highly accurate Quality of Service (QoS) measurements for V2X communication in practical field environments and to gather a set of delay measurements for a working CP system, accompanied by scalability discussions. The results provide a basis for evaluating the delay impact of single components and the applicability of CP use cases from the perspective of time advantage.

INDEX TERMS Autoware, automated driving, collective perception, cooperative connected automated mobility, ros, vehicle communication platform to anything, vehicle to anything.

I. INTRODUCTION

COLLECTIVE perception (CP) is part of cooperative connected automated mobility (CCAM) and intelligent transport systems (ITS). CP deals with sharing of sensor information. For example, an ITS station (ITS-S) provides sensor information, i.e., infrastructure sensor, vehicle, or similar. A Collective Perception Message (CPM) is then filled with this information and sent via vehicle-to-everything (V2X). Other ITS-Ss can then incorporate that perception information into their local perception system [1].

Detailed knowledge about the delays — from an observable event until the awareness of it within another ITS-S — is vital if one aims to discuss the applicability of CPMs for the safety of vulnerable road users (VRUs). Rauch et al. [2] were among the first to discuss V2X network delays, which Pilz et al. [3] extended by structuring the delays into perception, communication, and fusion. At the same time, Schiegg et al. [4] showed the performance of CP communication in a V2X simulation environment. Volk et al. [5] combined this research and provided details of the CP delay chain from a communication perspective extracted from a simulation. Cui et al. [6] provide an overview of existing research. However, details in the CP delay chain, such as communication cycle times, still need to be included and discussed.

This paper will analyze the delay chain of CP to extend the baseline for discussions of CP use cases, i.e., to discuss the applicability of CP for safety and comfort scenarios from a delay perspective. We aim to quantify CP delay parameters from a systematic point of view by analyzing the components of a live CP implementation. We aim to provide an overview of how much influence a CP delay element has on the overall delay. Discussion points are (i) the absolute amount of delay a CP element produces that can be measured from a working system, (ii) the amount of variable delay it creates, meaning the amount of delay that can be expected, and (iii) the limitations where CP becomes unfeasible. Expectations from the literature are compared to field-operational data collected from an automated driving system supporting CP. The implementation allows a low channel load quality-of-service (QoS) analysis of 802.11p with highly time-accurate

The review of this article was arranged by Associate Editor Stefania Santini.

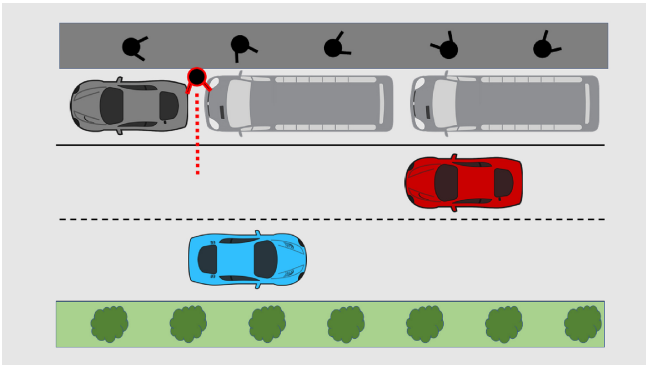

FIGURE 1. City side-by-side scenario: VRU (red circle) walking out between vehicles. Ego vehicle (red) can not see VRU. Other ITS-S, such as standstill vehicles (gray), other active vehicles (blue), or RSUs (not in the picture), can warn about the pedestrian.

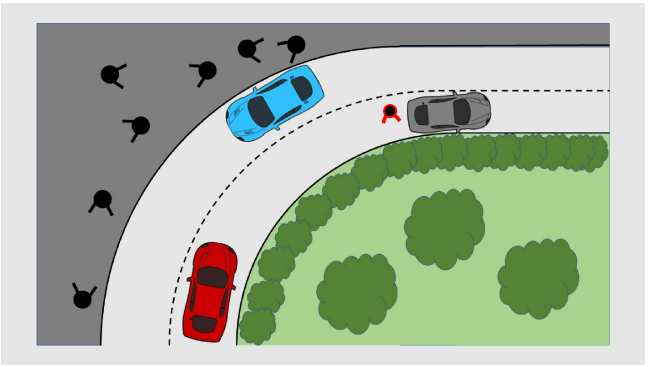

FIGURE 2. Highway scenario: VRU (red circle) behind standstill vehicle (gray). Ego vehicle (red) can not see standstill vehicle. Other ITS-S are aware of VRU and can send warnings via CP. Other ITS-S could be other active vehicles (blue), standstill vehicles (gray), and RSUs (not in the picture).

measurements, and a detailed analysis of perception and fusion components involved.

This paper will start by specifying scenarios in Section II to provide a frame for use cases. This paper will then describe how the delay information is gathered and calculated in Section III, followed by a detailed description of the delay components of CP in Section IV. Afterward, the measurement setups are presented in Section V, followed by the results in Section VI. In Section VII we elaborate on the scalability issues of V2X, before comparing the measured results to results from the literature in Section VIII. Finally, Section IX aligns our work with related research before concluding in Section X.

II. SCENARIOS

This paper aims to quantify delay parameters to discuss how CP can contribute to VRU safety and passenger comfort from a timing perspective. We are therefore looking at three scenarios to aid the decision process of using CP in similar situations. In our work, we assume high sensor accuracy and positional awareness of the automated vehicles to focus on the information delay.

In all cases, we compare the time available for decision-making of the ego vehicle in two scenarios: (i) with CP data made available via V2X and (ii) with data available via onboard perception. The questions are: (i) how can CP generate a benefit for safety and comfort? (ii) how much time benefit can CP generate compared to onboard perception?

A. CITY SIDE-BY-SIDE SCENARIO

The city side-by-side scenario, shown in Figure 1, describes our ego vehicle, shown in red, driving straight ahead. On the side of the road, there are parked vehicles. One of the parked vehicles is a bigger vehicle, such as a bus, that blocks the view of a VRU stepping onto the road. Without V2X, our ego vehicle can detect the VRU only when it is stepping on the drive path and into the sensor view. With V2X, an aware ITS-S could send out a CPM to tell our ego vehicle about the VRU. The aware ITS-S could be a standstill vehicle, another active vehicle shown in blue, or an RSU.

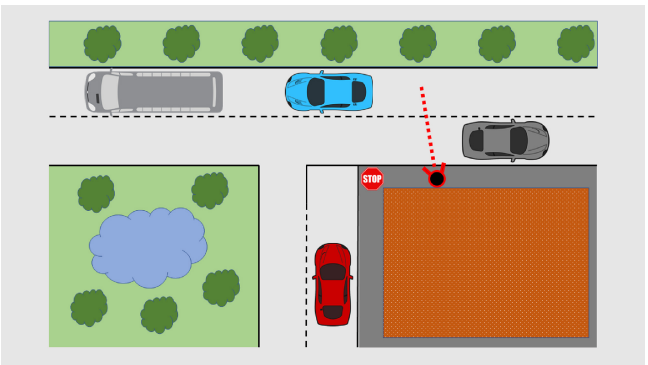

FIGURE 3. City intersection scenario: VRU (red circle) crossing street behind the corner. The ego vehicle (red) will be turning right at the intersection. Other active vehicles (blue) can warn about the situation.

B. HIGHWAY SCENARIO

The highway scenario is shown in Figure 2 and describes our ego vehicle, shown in red, driving on a highway around a corner. Around the corner is a broken-down vehicle, shown in gray, and a VRU standing on the road. Without V2X, the ego vehicle has to get far enough around the corner to detect the VRU with its sensors. With V2X, another ITS-S can tell the ego vehicle about the VRU. Other ITS-Ss could be the standstill vehicle, the other active vehicle shown in blue, or an RSU.

C. CITY INTERSECTION SCENARIO

The city intersection scenario is shown in Figure 3 and describes an ego vehicle, shown in red, that takes a right turn at the intersection after stopping at the stop sign. At the same time, a VRU may cross the road. Two things can happen here, depending on whether the VRU crosses the road: one possibility is that another ITS-S sends out free space via CP. The ego vehicle ignores the stop sign, improving performance. The second possibility is that the VRU crosses

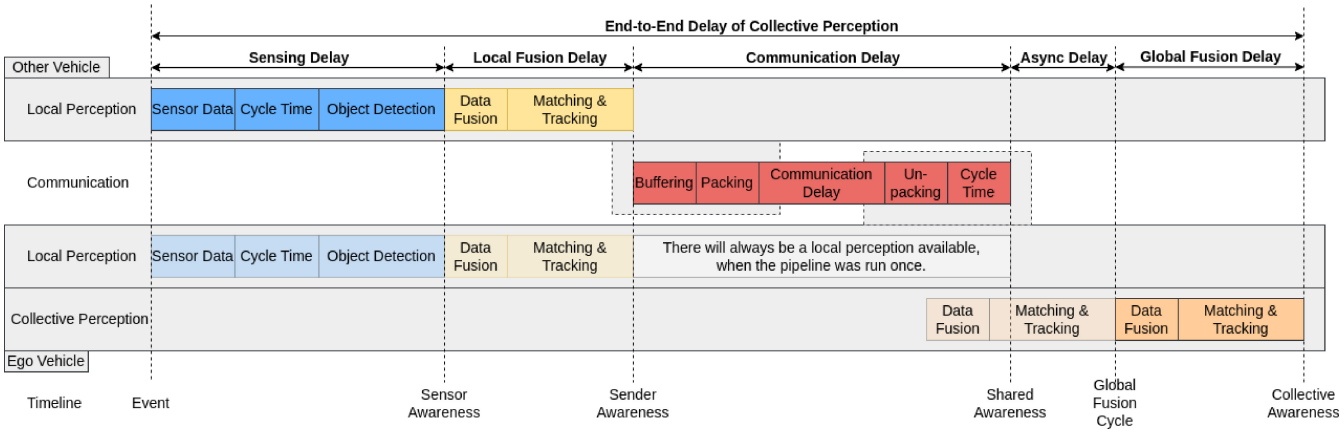

FIGURE 4. CPM delay chain: from an event happening until collective awareness. Involved are (i) the other vehicle and communication partner that senses its environment and sends the information via V2X and (ii) the ego vehicle that runs its local perception and puts the received global perception on top. The global fusion is run asynchronously to the received CP information.

the road, and the ego vehicle slows down in advance to increase comfort.

III. METHOD

This work aims to provide a systematic view of the delay of a working CP implementation in an automated vehicle. A systematic view requires three significant viewpoints: (i) the overall delay of information, meaning the time needed for information to be available in another ITS-S, via CP, after a triggered event, (ii) the V2X delay that can be expected in scenarios, such as described in Section II, and (iii) the limitations caused by components involved.

First, the delay components of the CP delay chain have to be defined. This is done by extending the preceding work of Volk et al. [5], and others [2], [3], [4], as mentioned in Section I. The core finding is a systematic representation of the CP delay components, with expected results from the literature, as presented in Section IV.

Next, we analyze the quality-of-service (QoS) of V2X communication in low channel load scenarios. We thereby extend works, such as of Böhm et al. [7] and Altinel et al. [8], who studied the connectivity and link quality of ITSG5, by a study on the end-to-end (E2E) delay of an implemented system. Our results present an expectation for scenarios as defined in Section II. The measurements are executed in an environment with low V2X channel load, with highly time-synced platforms and software designed for QoS measurements, as will be described in Section V.

Afterward, we present the implementation of a complete CP chain into our automated driving demonstrators (ADDs), which are street-legal vehicles controlled via a drive-by-wire system and a software stack that we present in Section V.

Finally, we aggregate the measurement results in Section VI and discuss in Section VIII our findings in comparison to the delay components we expected from the literature in Section IV.

IV. DELAY COMPONENTS

Rauch et al. [2] were the first to show V2X-related delays in a diagram. Pilz et al. [3] then suggested extending those times and splitting them into three major components: (i) the sensing-related delay, (ii) the communication-related delay, and (iii) the fusion-related delay. The CPM delay time is the sum of delays of all components, as shown in Equation (1).

$$t_{cpm} = t_{sensing} + t_{communication} + t_{fusion} \tag{1}$$

The three components can be split into more complex delay components, as shown by Volk et al. [5], focusing on V2X communication. Hence based on both Rauch et al. [2] and Volk et al. [5], Figure 4 shows the CP delay chain with four significant changes: (i) it takes into account sensor processing times, (ii) it adds a necessary buffer on the receiving end, (iii) it removes the reference of local perception of the receiver, as the receiver always has a local perception available, and (iv) it shifts the focus from the communication to perception and fusion. Shifting the focus to perception and fusion is done, as this paper views communication as one complete transmission block. In other words, this communication is a complete message-passing system with QoS parameters.

The following sections will discuss the (sub)components of the delay chain, with aid from related research. Related research also provides delay expectations aggregated in Table 1. In a later step, results from the live implementation are integrated into this Table 1. Section VIII will then compare the expectation to the measurements of the working implementation.

A. SENSING DELAY

First, an *event* triggers the sensing phase of the CPM chain. Let this event be a pedestrian that walks into the field of view of a sensor, i.e., when it can be detected the first time. The sensor now adds (i) its data delay, (ii) its cycle delay, and (iii) its related object detection delay.

TABLE 1. Components of delay with reference to Figure 4. The delay components are derived in Section IV, with estimates from the literature. Measurements are provided by the results in Section VI.

Delay Component	Delay Description	Estimate from literature [ms]	Measured [ms]
Sensing Delay	sensing delay	**206-280**	**207.7**
|- Sensor Data Delay	from event to sensor image	camera: 25 lidar: 50	camera: 24.1 lidar: 34.3
|- Cycle Time	from sensor image to available data	camera: 25-100 lidar: 50-100	camera: 62.5 lidar: 100
|- Object Detection	from sensor data to object data	56-130	camera: 69.0 lidar: 73.4
Communication Delay	from collecting to redistribution	**11-254** +[100-∞]*	**19.1/119.1/419.1** +[100-∞]**
|- Buffering	fastest (ETSI): 10Hz	2-100	1
|- Packing	ROS2 and packing	2	1.1
|- Transmission Delay QoS	C-ITS load low C-ITS load medium*'** C-ITS load high*'**	3-50 100-500* 500-∞*	5/105/405 exp. +[100-500]** exp. +[500-∞]**
|- Unpacking	unpacking and ROS2	2	11
|- Distribution	sensor like: 10Hz	2-100	1
Fusion Delay	local sender and global receiver	**4-6**	**104.8**
|- Sender (Local) Fusion Delay	local multi-sensor fusion	2	2.4
|- Asynchronous Fusion Delay	local multi-sensor fusion	0-2	100 (w.c.)
|- Receiver (Global) Fusion Delay	global multi-sensor fusion	2	2.4
CP Delay Time	**all in***'**	**221-540** +[100-∞]*	**331.6/431.6/731.6** +[100-∞]**

* Scalability (ITSG5): delay may be drastically increased by the Access Layer due to channel load. (Section IV.B.3).

** Scalability (ITSG5): expectations for Access Layer delays. Test results are gathered from low channel load environment. (Section VI.A.3).

1) SENSING: DATA DELAY

The data delay is dependent on the sensor type. For cameras, the data delay consists of the frame acquisition time and the frame read time [9], as shown in Equation (2). For lidars and radars, similar principles apply.

$$t_{data} = t_{acquisition} + t_{readout} \tag{2}$$

The data delay can be found in the datasheets of the sensors. If it is included, the sensor data delay is usually close to the highest frequency of the sensor cycle time. The reason is simple: the vendor wants to provide as high of a frequency of sensor data as possible without providing redundant sensor images. So the cycle time's lower bound is the data delay's upper bound.

To calculate expectations for the data delay, we look at the sensor cycle time. For a state-of-the-art lidar, sensor [10] this is $10-20Hz$. For a state-of-the-art high-resolution camera [11], this is $10-40Hz$. In both cases, we expect the upper limit of the frequency to be close to the data delay itself. Higher frequencies are mostly not possible due to data acquisition limitations. Hence worst case, the data delay is $1/f_{sensor,max}$, which results in $25ms$ for a camera and $50ms$ for a lidar.

2) SENSING: CYCLE TIME

The cycle time of the sensors is calculated from the cycle frequency, as mentioned above. This leads to cycle delays of $25-100ms$ and $50-100ms$ for state-of-the-art cameras and lidars, respectively. The best case is receiving the sensor data before a new cycle. The worst case is having the sensor data ready but missing the cycle. Hence, in the worst case, one missed cycle has to be estimated. In summary, the sensor delay is the sum of the data delay and the cycle time, as shown in Equation (3).

$$t_{sensor} = t_{data} + t_{cycle} \tag{3}$$

3) SENSING: OBJECT DETECTION DELAY

The object detection delay is created by extracting objects from the sensor data to be able to transmit them as CPMs. Starting with the raw sensor data, it is possible to (i) proceed directly with object detection or (ii) proceed with sensor data fusion first and then do object detection. Pilz et al. [3] point out that dependencies between elements of the CPM influence the approach. For example, fusing raw sensor data will generate better object detection results, but detecting objects first and fusing them later is faster. Also, Hackett and Shah [12] show that the fusion of raw data of multi-sensor systems increases accuracy but requires a pre-processing step that introduces dependencies between sensors. These dependencies can be avoided by doing object detection first and fusing later. On the receiver side, the fusion of sensor data from multiple onboard sensors, i.e., camera, lidar, and radar, is also up to the respective implementation. In this work, both the sender and receiver will follow the object detection first and fusion later approach. This approach means that object detection is done directly after pre-processing sensor data and before the

fusion of objects. Back to the sole object detection, state-of-the-art object detection algorithms can analyze around $7.7 - 17.8FPS$, as discussed by Wen and Jo [13], which is a $56 - 130ms$ delay for object detection.

4) Σ SENSING DELAY

The overall time needed for sensing includes the above-stated sensor data delay, sensor cycle time, and object detection delay, as shown in Equation (4). Hence, given the information stated, this estimates a best/worst case of $81 - 280ms$ for the sensing delay, as shown in Table 1.

$$t_{sensing} = t_{data} + t_{cycle} + \left(t_{class\_fusion}\right) + t_{object\_detection} \quad (4)$$

B. COMMUNICATION DELAY

The communication delay consists of several elements. For this paper, we focus on three major categories: (i) the buffering and distribution, which are the most influential cycle times, (ii) the packing and unpacking, which are necessary data conversion times, and (iii) the transmission of data as a whole, because it can be defined with QoS parameters.

1) COMMUNICATION: BUFFERING AND DISTRIBUTION

By specification [1], the CPM frequency has to be between 1-10Hz. Sending CPMs immediately would be better for the delay. However, related literature [14], [15], [16] shows that this will pollute the channel. The ETSI CPM standard suggests 10Hz sending for new objects and objects with a high dynamic, while 1Hz is enough for static objects or objects with a straightforward prediction. In this paper, we look at the fastest scenario with 10Hz. We expect, for example, an object to appear or to change its direction in a critical, unexpected way. This then results in a maximum time of $100ms$. The minimum cycle time is more complex: if the local fusion frequency is 10Hz, the buffering could be coupled with the output of the local fusion, which is typically also 10Hz. In that case, the buffering delay is only the time necessary for analyzing the object data. If one uses a simple comparison algorithm, this can be done $< 1ms$. Including the DDS transport time $< 1ms$, the overall buffering time is between $2 - 100ms$.

On the receiver side, the CPMs must be buffered before distribution. As discussed above, typical sensor distribution frequencies are 10Hz; hence, 10Hz distribution frequency is also reasonable for V2X data. The best case is that we receive the data and can immediately distribute the data, which would again result in $< 1ms$ processing time. For a systemic approach, we also include some internal transport time. In the case of ROS2, the internal transport time of DDS is $< 1ms$ for a single transmission, provided by Kronauer et al. [17]. All in all, the distribution time is then $2-100ms$. However, one should always expect higher delays for the implementation, as the sender and receiver are not synchronized.

2) COMMUNICATION: PACKING AND UNPACKING

There is a delay in packing and unpacking objects. As V2X is a low-latency transmission technology, it is essential to add delays, such as packing and unpacking, to the systematic approach. For our implementation, we expect that ROS2 objects are packed into transmittable ASN1 bitstreams and vice-versa. This is again the internal transport time of DDS with $< 1ms$ [17] for a single transmission. The packing and unpacking of ASN1 streams to C arrays and their conversion from and to ROS2 messages are estimated to take $< 1ms$. This results in $2ms$ packing and unpacking time, respectively.

3) COMMUNICATION: TRANSMISSION DELAY

Much research has already been done (i) by analyzing V2X to confirm the low latency transmission times [18], [19], [20] in perfect side-by-side scenarios, (ii) by analyzing V2X congestion control mechanisms [21], [22] in heavy traffic scenarios, (iii) by providing simulation results for packet-loss [23], [24], [25], and (iv) by providing field measurements for connectivity [7], [8]. However, the QoS of wireless technologies, such as V2X is complex. The following paragraphs aim to provide an estimate for scenarios as defined in Section II.

The physical layer transmission times of all four existing V2X standards have a comparable delay. Anwar et al. [18] specify the 802.11p physical layer transmission time from its standard as $0.344ms$ for 100 bytes and $4.08ms$ for 1500 bytes. Transmission delays of 802.11bd, LTE-V2X, and 5G-NR-V2X are in a similar range with $0.336/3.98, 1/11, 0.75/8$ all in ms for 100/1500 bytes. However, other communication layers add a delay on top of that. Campolo et al. [19] point out that a decentralized environmental notification message (DENM) requires around $3ms$ E2E in an analysis of LTE-V2X as the lower boundary.

To get an upper estimate for low channel load scenarios, Correia et al. [26] tested CPM sending with 10-75 vehicles as CPM payload and found message delays between $0 - 45ms$. The $0ms$ is likely due to internal processing and time synchronization only via network time protocol (NTP). Considering measurement errors of up to $5ms$ through NTP, an upper boundary of $50ms$ seems fitting. For ITSG5 it has to be stressed that the expected transmission delay of $3 - 50ms$ for one message is only a frame for the minimum and the maximum expected transmission time in low channel load scenarios.

When at least one other transmitting ITS-S occupies the V2X channels, there are busy times and collisions in transmissions. The channel load influences the transmission delay, caused by necessary QoS functionality defined in the Medium Access Control (MAC) layer and its extensions [27]. The complexity and scalability of V2X transmission, with a focus on ITSG5, is outlined in Section VII. Here, we aim for a rough estimate. Shah et al. [28] show the relation between delay caused by the MAC layer and the number of vehicles, respectively their differential speed. Simulation results show a near-linear behavior for the delay

of non-safety data. Depending on several parameters, the average packet delay is $100ms$, for 25 vehicles. A similar delay is caused by a $35km/h$ differential speed.

Zheng and Wu [29] show results depending on the enhanced distributed channel access (EDCA) queue. Depending on the messages, their priority, their payload, their distribution frequency, and the overall number of ITS-S, a single message may suffer from delays $>> 100ms$. This means that the amount of delay the CPM might have in loaded channels can not be specified in a trivial form. Worst case, the message might never arrive. Either due to the MAC delay or due to packet loss, caused by obstructions or out-of-range communication. Details are discussed in Section VII.

The transmission delay can be expected to be around $5-50ms$ in low channel load scenarios. For medium load with 25-100 vehicles, one can take the simulation results of Shah et al. [28] with $100-500ms$. With a high channel load, the delay is unbounded without guaranteed delivery, hence one can expect $500-\infty ms$. One may argue about the definite delays for the combination of channel load and differential speed. But for a rough estimate, the given frames should be fitting.

4) Σ COMMUNICATION DELAY

The final communication delay is the sum of all five mentioned communication delay subcomponents, as shown in Equation (5). The final communication delay can be calculated to $11-254ms$, for low channel load, as shown in Table 4. For medium and high channel load, one can expect at least $100ms$ of additional delay, depending on the circumstances of transmission, as discussed in Section IV-B3.

$$t_{comm} = t_{buffer} + t_{pack} + t_{transmit} + t_{unpack} + t_{distribute} \quad (5)$$

C. FUSION

Data Fusion can be separated into delays for data fusion and matching & tracking for the fusion of high-level object data, as shown in Equation (6). In a multi-sensor setup with high-level object data, data fusion fuses the object data from the object detection step. Matching is then needed to find the corresponding objects. Finally, the tracking step produces tracks of the matched objects.

$$t_{fusion\_components} = t_{data\_fusion} + t_{matching\_tracking} \quad (6)$$

1) FUSION: DATA FUSION, MATCHING AND TRACKING

For a detailed analysis, it makes sense to analyze the fusion components as stated in Equation (6). However, state-of-the-art systems, such as Autoware [30], provide an all-in-one solution for matching, tracking, and object data fusion as list and matrix operations, as well as Kalman filters. Hence one has to compare the overall inference time of such solutions, as suggested by Jahromi et al. [31]. Jahromi et al. analyzed state-of-the-art algorithms for the complete detection and object data fusion pipeline and found inference times of $92-185ms$. The elements matching, tracking, and fusion are only parts of this time. As Jahromi et al. [31] provide the times for detection and fusion, we could subtract the time for object detection, see above Wen and Jo [13], from that. However, this only gives a rough idea of around $30ms$ of overhead when subtracting the minima and maxima without discussing different hardware and algorithm setups. A better estimate is to look at what matching, tracking, and object data fusion are processing. Matching and tracking are the nearest neighbor search which can be done in around $1ms$ for a few objects, according to Gallego et al. [32], while object data fusion is a simple merging of lists, which is even faster. Hence we will estimate a rough fusion time of $2-30ms$.

2) FUSION: LOCAL, GLOBAL AND ASYNC DELAY

Next, as shown in Figure 1, two sections have data fusion times. The first one is the local fusion for the sender. The second one is the global fusion of the receiver. Two main approaches can integrate CP data: fusing everything at once, meaning fusing the local data and the global CP data directly every time. However, as elaborated by Pilz et al. [3], data freshness, cleanliness, and security issues will arise. Without external moderation, local data should be the reference. Else, global data may corrupt the CP. Hence, the second approach is that the receiver does local fusion first and then global fusion. From the perspective of delay, it is best to use the most recent local fusion to integrate the global fusion, as is shown in Figure 4. This way, perception information is already here.

In conclusion, the fusion time consists of the local fusion time of the sender, the asynchronous delay of the global fusion cycle, and the global fusion time of the receiver, as shown in Equation (7) and Figure 4.

$$t_{fusion} = t_{local\_sender} + t_{async} + t_{global\_receiver} \quad (7)$$

However, the global fusion itself has a catch. Figure 4 shows that the event starts the timeline. This means the currently active local perception of the Ego vehicle may be already more recent than the CP data received from the other vehicle. In this case, the global fusion has to consider the timing offset of the CP data to the newer local perception data. Generally, one can implement two strategies: either alter the CP data and, for example, predict it into the future or alter the new local perception to fit the old CP data. The more accurate way is to do the latter; hence we also estimate for the global fusion to do one fusion step with older data and then add the new local perception if it was already available. In summary, there will be one global fusion step if the local perception fits but two fusion steps if the local perception is new.

3) Σ FUSION DELAY

The delay results of Equation (7) will be most accurate when calculating local and global delays separately. A separate calculation is essential when two ITS-S highly differ

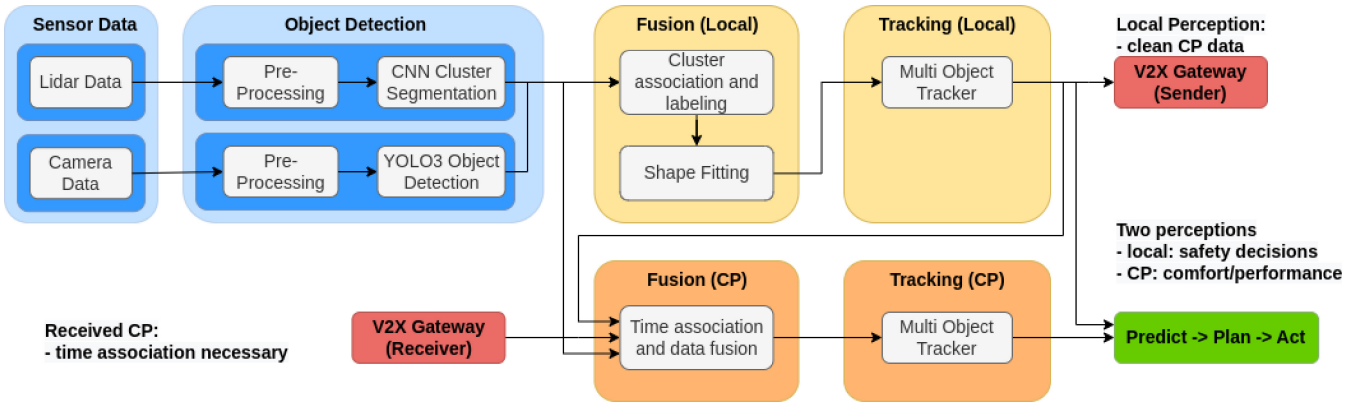

FIGURE 5. Perception part of the automated driving software stack. The diagram shows the flow of information from sensor information to the planner and where the V2X communication is connected. The software is based on the architecture proposal of Autoware [30].

in performance or with many objects. However, related research, such as Correia et al. [26], estimate below 75 objects per intersection, which is not yet computationally expensive, as shown by Gallego et al. [32]. For scenarios with few objects, as discussed in Section II, we can simplify the calculations as shown in Equation (8). This equation calculates the final fusion time of $4-6ms$ as seen in Table 1.

$$t_{fusion} = 2 * t_{fusion\_components} + \left(t_{fusion\_components}\right) \tag{8}$$

V. SETUP

The CP domain consists of many established fields. Most measurements are obtained from simulations or done in separate environments. Fortunately, we have a handful of street-legal automated driving demonstrators (ADDs). For this work, we equipped two of them with V2X and CPM capability to investigate the delay elements of the entire CP chain. Additionally, we provide QoS measurements for 802.11p, as specified in Section II. The following Sections will describe the setup of our vehicles and the used V2X communication platform.

A. AUTOMATED DRIVING DEMONSTRATORS (ADDS)

At the Virtual Vehicle Research GmbH,[1] we share a handful of ADDs across projects. Each ADD has a different setup for hardware and software. This paper used the Ford Fusion and Ford Mondeo with their respective hardware setup. Both vehicles can run the same software stack. The software stack currently builds upon the Autoware Architecture Proposal. However, we are already in the process of upgrading it to Autoware Universe. The software stack and the hardware configuration of the two used ADDs are described in the following sections.

1) AUTOMATED DRIVING SOFTWARE STACK

The current software setup is based on Autoware [30]. Autoware provides a complete software stack and functions for automated driving based on ROS. It contains modules that implement the sens-plan-act principle. The Autoware stack provides detailed documentation for the software stack, including setup and tutorials. It then only needs to be configured for the particular sensor setup in the vehicle and the drive-by-wire system.

To be more specific, the Autoware architecture proposal[2] is used, which is a work-in-progress architecture for a next-generation of Autoware. The Autoware architecture proposal was proposed to the Autoware Foundation and was migrated to the Autoware Universe software stack[3] in a recent step. For this paper, we must state that Autoware Universe comes with partly more optimized implementations in the perception chain. Still, the processing chain's general principle holds, making it comparable. For simplicity, only the term Autoware is used in the course of further work.

For this paper, only the perception part is relevant. Figure 5 shows the data processing diagram, whereas each of the colored boxes represents a separate ROS node. The sensor data nodes are drivers acquiring sensor data and providing the raw data to the ROS environment. The preprocessing reformats the data to be used by the Autoware object detection implementations. The lidar object detection uses a convolutional neural network (CNN), which outputs 3D clusters representing detected objects. The camera object detection uses YOLO3 to output labeled 2D objects. The local fusion then does cluster association to fuse the 3D and 2D objects. The associated objects are then shape-fitted to the detected object class, resulting in bounding boxes. Afterward, the multi-object tracker matches the detected objects with previous detections using a Kalman-Filter-based approach and generates a track for each object. The planner will then predict future tracks and act accordingly. At the same time, the V2X gateway will process the data to be sent to another ITS-S.

On the receiving end, the V2X gateway receives the object data. The receiving vehicle also ran the same processing as

1. https://v2c2.at

2. https://github.com/tier4/AutowareArchitectureProposal.proj
3. https://github.com/autowarefoundation/autoware.universe

TABLE 2. Hardware setup — ADDs.

Type	Ford Fusion	Ford Mondeo
CPU	Intel i7-9700K	Intel i7-11800H
GPU	Nvidia RTX 2070	Nvidia RTX 3070 Mobile/Max-Q
RAM	32GB DDR4	32GB DDR4

the sending vehicle. Hence a local fusion is available. The CP fusion first does the time association of the incoming CP data. The time association checks the timestamp of the incoming message to associate it with the correct local fusion, which can either be the most recent fusion or the previous one. The previous one is used if the local sensor data is newer than the CP data and has already been fused. In this case, the time association first manages the fusion of the CP with the local fusion and then adds the most recent sensor data. The CP tracking finally does the same as the local tracking. It matches the objects with the already known objects and produces the track. The planner now receives tracks for local and CP objects. The separation allows basing decisions on the source of information. I.e., internal policies require us to base critical safety decisions on local perception only. We may not trust external sources, except if we are on a proving ground.

2) ADD: FORD FUSION

The Ford Fusion, shown in Figure 6, is equipped with four lidars, an Ouster OS2-128 and three Ouster OS1-64, as well as two cameras, both of which are FLIR Blackfly S. The Ford Fusion is currently the ADD with the most sensors in the carpool of the Virtual Vehicle Research GmbH. The computing hardware is three-year-old mid-range consumer hardware, as shown in Table 2. This vehicle will be the ITS-S that generates CPMs from the local perception.

3) ADD: FORD MONDEO

The Ford Mondeo is equipped with different sensors, depending on the use case. The computing hardware is an Alienware laptop, with specifications shown in Table 2. For this paper, we use the global navigation satellite system (GNSS) with correction data for $< 10cm$ positional accuracy. This positional information is sufficient to test the performance of the receiving software stack. The Ford Mondeo is in the role of the ego vehicle that receives CP data. In other words, the local perception will only consist of the GNSS localization to visualize the CP. In all other means, the same software stack applies as shown in Figure 5.

B. COMMUNICATION PLATFORM

The vehicle communication platform to anything (vehicleCAPTAIN) is the V2X communication relay in this paper. The core software components are available as part of the FOSS vehicleCAPTAIN toolbox [33]. The primary purpose of the vehicle_captain_routing_core is to provide a single platform to host various V2X communication modules and

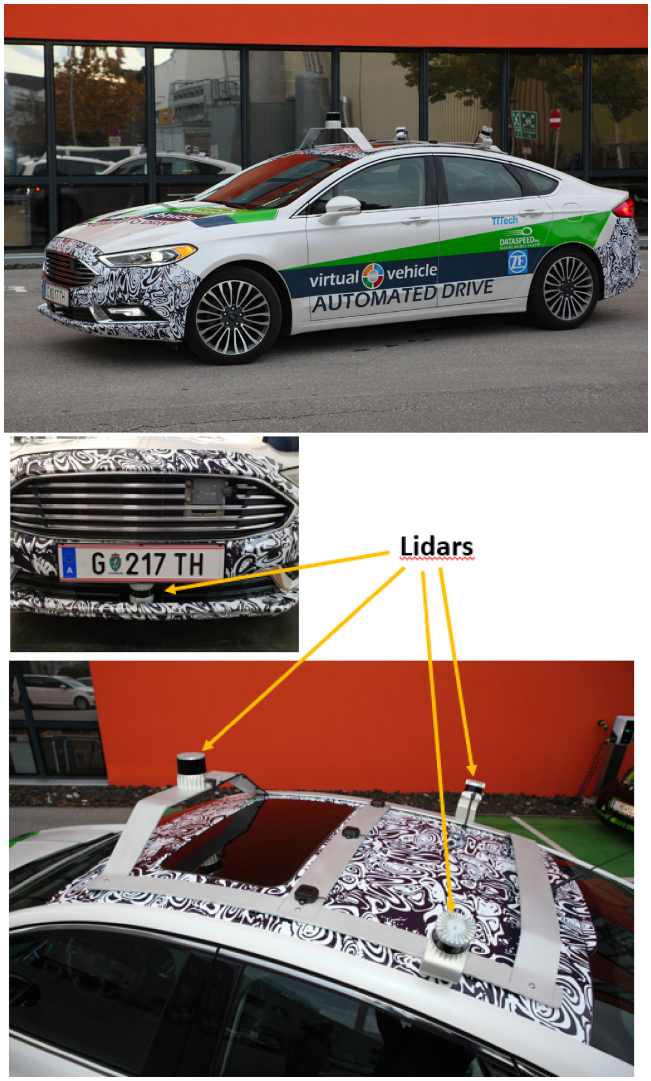

FIGURE 6. Virtual Vehicle ADD: Ford Fusion, with sensor setup, mounted on the roof.

control them via a single software interface. In other words: there is one software interface to receive and send V2X messages. The vehicleCAPTAIN controls the V2X modules via their APIs. The user does not have to deal with implementing a specific module. This way, the user does not have to deal with the underlying communication layers. The physical layer could be ITSG5, C-V2X, IP-based, or any other interface one specifies.

The message handling is also done by software in the vehicleCAPTAIN toolbox [33]: (i) the vehicle_captain_its_lib_c_cxx is software for decoding/encoding of ASN.1 bitstreams, (ii) the v2x_msgs are the ROS2 pendant to the ETSI specified ASN.1 messages, and (iii) the v2x_gw is a ROS2 node that translates between ASN.1 bitstreams and ROS2 type V2X messages, by connecting to the vehicle_captain_routing_core and by making use of the libraries above.

For this paper, the vehicleCAPTAIN hardware is equipped with one Unex SOM-301E to guarantee the low latency of

TABLE 3. Hardware setup — vehicleCAPTAIN.

Component	Type
Base Board	Raspberry Pi 4 8GB
V2X Module	Unex SOM301-E
GNSS Module	ublox ZED-F9P

V2X. The Unex SOM-301E ships with V2XCast.[4] V2XCast takes care of the V2X stack during transmission and sending. The vehicle CAPTAIN is mainly an abstraction so that one can exchange the underlying sending hardware and software without changing the implementation of the V2X communication interface in the ADDs and other demonstrators.

The vehicleCAPTAIN hardware distinguishes this work from similar approaches of delay measurements by synchronizing the hardware clocks with 1pps. The hardware, listed in Table 3, allows the synchronization of sender and receiver to be good enough for the low latencies of V2X. Combined with Qosium, a QoS analyzer software, we can do a comprehensive and highly time-accurate analysis of V2X QoS in field-operational situations.

1) MEASUREMENT RESOLUTION FOR QOS

The QoS measurements necessitate high-precision clock synchronization between measurement points. The expected transmission delays are within a few milliseconds. These low transmission delays require the clock synchronization accuracy to be substantially below one millisecond throughout the measurements. Network Time Protocol (NTP) cannot reach sufficient synchronization accuracy for our study. Typically, the obtained accuracy of NTP is about a millisecond at best. Precision Time Protocol (PTP) allows reaching our needs. However, PTP needs wired synchronization channels to reach this accuracy, which is impossible to realize in mobile measurement scenarios. Thus, we base our approach for clock synchronization on the one Pulse Per Second (1PPS) signal obtained from a GNSS. Using 1PPS enabled us to reach below $100\mu s$ synchronicity with the following setup: we use gpsd, a GNSS service daemon, in combination with chrony, an NTP implementation supporting GNSS reference sources. The 1pps source provided by the ublox GNSS module allows synchronization of the host device's clock without being directly connected.

We verified the synchronicity of the setup, as shown in Figure 7: the UART interface connects two vehicleCAPTAINs. One vehicleCAPTAIN triggers the verification measurement by sending data from its Tx pin. Both vehicleCAPTAINs receive the same data simultaneously on the RX pin. Both devices store the current timestamp when receiving the trigger signal, synchronized via 1pps. Comparing these timestamps results in the offset between the devices, as shown in Figure 8. From the data, one can see that the synchronicity is below $10\mu s$ for most samples, with spikes

4. https://unex.com.tw/v2xcast/

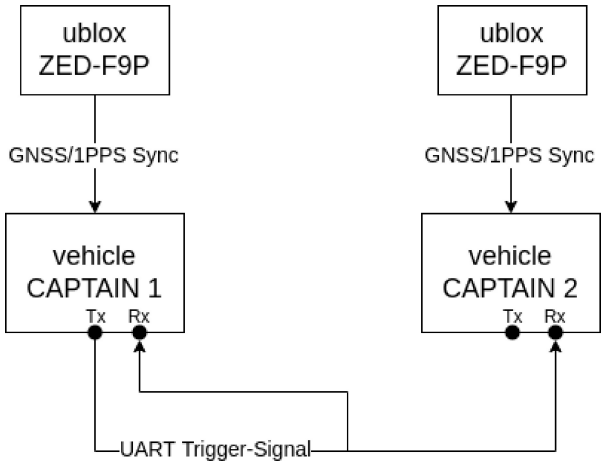

FIGURE 7. Setup to determine the time offset between 2 vehicleCAPTAINs.

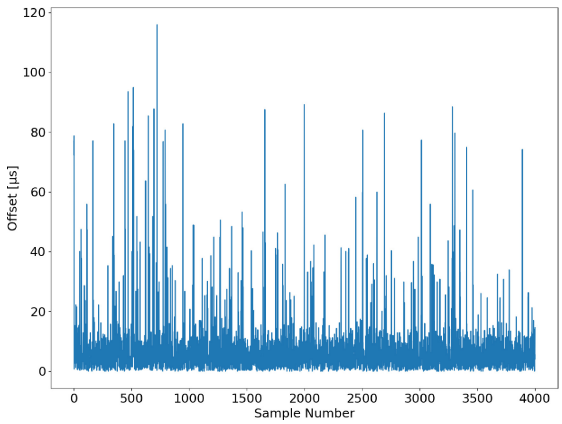

FIGURE 8. Box plot of the offset between two vehicleCAPTAINs, during time synchronization evaluation.

TABLE 4. Significant metrics of the time offset between 2 vehicleCAPTAINs, with 1pps synchronization.

Metric	Value [μs]
minimum offset	0.0
maximum offset	115.871
average offset	6.967
median	4.768
interquartile range	5.722

below $100\mu s$. Outliers of more than $100\mu s$ might result from the software overhead of the evaluation method. However, chrony deals with such spikes, and as seen from the analysis in Table 4, the synchronicity is higher than the expected transmission times and, therefore, suitable for highly accurate measurements.

2) QOSIUM QOS MEASUREMENT SOFTWARE

The minimum delay of V2X communications, especially the physical and MAC layers, can be calculated from specifications, as elaborated in Section IV. Other QoS parameters, such as packet loss, have been discussed and simulated in [19], [20], [34], [35]. However, practice and theory come

TABLE 5. Qosium Delay test results, for one sending ITS-S. Averaging interval (T) = 1000ms. Packet E2E delay (τ), for one sending ITS-S.

Type (Distance)	$\sum \overline{T}$	$packetloss_{\forall}[\%]$	$\tau_{min,\forall}[ms]$	$\tau_{max,\forall}[ms]$	$\overline{\tau}_{\overline{T}}[ms]$	$\widetilde{\tau}_{\overline{T}}[ms]$	$\hat{\sigma}^2_{\tau_{\overline{T}}}[ms]$	$\overline{jitter}_{\overline{T}}[ms]$
Campus Standstill ($10m$)	173	0.12	3.735	13.611	4.477	4.304	0.473	1.656
Campus Round ($50m$)	259	13.69	3.731	16.612	4.547	4.311	0.582	1.619
Campus Round ($150m$)	266	55.74	3.823	67.258	4.619	4.391	0.703	2.118
Campus Round ($250m$)	278	57.78	3.792	59.842	4.641	4.378	0.765	2.444
City Follower ($50-100m$)	877	3.71	3.694	141.655	4.791	4.285	1.684	4.711
Autobahn Overtaking (max. $3000m$)	800	79.94	3.649	14.649	4.494	4.280	0.547	1.604

with a gap, especially in wireless communications. Also, simulation models need approximations and assumptions for many factors affecting wireless communications. There are no publicly available studies for highly accurate and comprehensive QoS measurements for V2X, which are essential when talking about performance and especially safety. For CP transmission in general and specialized CP use cases in specific, especially virtual towing [36], it is essential to have a constant flow of messages and a low delay for critical messages. With Qosium,[5] we can empirically measure an extensive amount of QoS statistics to extend the basis for discussing such parameters.

Qosium is a passive QoS measurement software. It measures traffic, flow, and QoS statistics in real-time for existing networks without injecting artificial test traffic for measurement. Qosium does not need any support from the applications and does not interfere with data transfer. The measurement solution does not have limitations on applications for measuring or network technologies over which the measurement is conducted. The statistics are one-way, which means that both directions are measured separately and simultaneously.

The measurement agents, called Qosium Probes, are installed onto two vehicleCAPTAINs for our evaluations. This setup allowed us to measure QoS statistics, such as delay, jitter, packet loss, and packet loss bursts (PLBs). PLB refers to sets of consecutively lost packets. However, as discussed in the following, the probes are not connected directly to the V2X communication interfaces but to the ethernet interface of the vehicleCAPTAIN hardware. Hence, results do not include specific parameters, such as RSSI values. The analysis is carried out for E2E packets between both platforms.

C. MEASUREMENT SETUPS

We split our measurements into two measurement campaigns. The first one is specifically for the V2X QoS measurements, where we had to synchronize the V2X communication platforms. The second one focuses on the full implementation of CP.

1) QOS MEASUREMENT SETUP

Our test setup equipped two vehicles with vehicleCAPTAINs, as shown in Figure 9. The first vehicle is our ADD

5. https://www.kaitotek.com

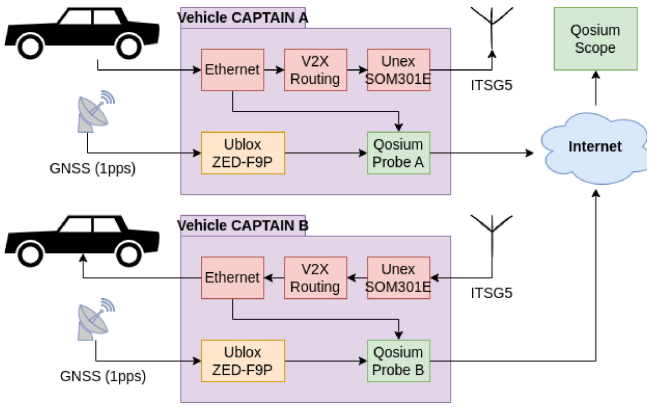

FIGURE 9. Test setup for 1pps synced QoS measurements with Qosium.

Ford Fusion. We connected vehicleCAPTAIN A via Ethernet to the automated driving system. The system creates cooperative awareness messages (CAMs) and sends them out at 10Hz. vehicleCAPTAIN A listens for V2X messages on its interface. In this case, vehicleCAPTAIN A will receive a CAM. The Qosium probe also registers this CAM on the Ethernet interface. vehicleCAPTAIN A then takes the CAM, processes it, and sends it out via a *Unex SOM301-E* interface. The V2X interface transmits the data via V2X to the other side, where another *Unex SOM301-E* interface receives the CAM. vehicleCAPTAIN B then collects the CAM from the module, processes it internally, and puts it out to be received in the other vehicle. There, a straightforward logger program collects the messages to avoid processing delays. During the Ethernet transmission, the second Qosium Probe detects those sent CAMs at the ethernet interface of vehicleCAPTAIN B and closes the measurement loop.

Qosium Scope processes the information from both ends and puts out QoS parameters. The packet matching is done by analyzing the contents of the TCP packets collected from the Ethernet interfaces of the vehicleCAPTAINs. By connecting the Qosium probes to the Ethernet interfaces instead of the V2X interfaces, we can also include the total processing overhead of the V2X network management in the E2E delay.

We chose six scenarios for the test drives, as listed in Table 5 and 6. In the first scenario, we left the vehicles standing in front of our office building to get reference data for close distances. The second, third, and fourth scenarios are city scenarios: we drove around the Inffeld Campus of

TABLE 6. Connectivity results with Qosium, for one sending ITS-S. Packet loss burst (PLB). Packet loss burst length (ℓ). Packet loss burst time (t).

Type (Distance)	$\sum packets$	$\sum PLB$	ℓ_{max}	$\bar{\ell}$	$\hat{\sigma}^2_{\ell}$	$t_{PLB,max}[s]$	$\bar{t}_{PLB}[ms]$	$\hat{\sigma}^2_{\bar{t}_{PLB}}[ms]$
Campus Standstill ($10m$)	1719	1	1	1	0	0.1	101	0
Campus Round ($50m$)	2574	175	12	1.88	1.88	1.2	86	160
Campus Round ($150m$)	2644	179	547	10.77	57.07	55.0	411	3539
Campus Round ($250m$)	2763	106	370	13.01	58.18	37.2	318	2908
City Follower ($50-100m$)	8719	140	8	1.76	1.30	0.8	29	85
Autobahn Overtaking (max. $3000m$)	7954	65	2420	67.87	391.95	243.4	554	11226

the Graz University of Technology (TU Graz). Here a block of buildings spans around $100m \times 250m$, which allows distances between the vehicles of $50m$, $150m$, and $250m$. Hence we got a good combination of free line of sight, one corner, and two corners between the vehicles. Scenario five was then driven from the Inffeld Campus to the highway through the urban area of Graz. Finally, scenario six is a highway scenario, where the leading vehicle started on the ramp, the second one waited for $2km$ and then followed with a differential speed of $40km/h$.

2) CP MEASUREMENT SETUP

To test CP in scenarios as specified in Section II, we equipped our ADDs Ford Fusion and Ford Mondeo with the V2X communication platform vehicleCAPTAIN, details in Section V. The Ford Fusion has four Ouster OS-1 lidars, and a stereo camera setup with two FLIR Blackfly S. The lidars provide raw point clouds and the cameras raw images. Both data streams are processed by the Autoware architecture proposal stack [30]. The Autoware components provide pre-processing of sensor data, object detection, and fusion of the object data. The fused data is then (i) used as an onboard sensing reference for the planner, (ii) used as input for the V2X communication, and (iii) used as a reference sensor input for the global fusion. With this form of implementation, a local fusion is always available where the global fusion can be put on top. Also, an automated vehicle planner can focus more on local fusion as a reference point. Moreover, the local fusion is not altered by global information. For the measurements, we chose to take a full round around the TU Graz Campus Inffeld, as this provides a good presence of vehicles and pedestrians, as one would see in a city scenario. From the rosbag, we are then able to extract delay statistics.

VI. RESULTS

We provide two kinds of test results: (i) a highly time-accurate field operational analysis of QoS parameters of ITSG5 in low channel load scenarios and (ii) a detailed field operational analysis of the components of the CP delay chain. The results are put directly into Table 1 to provide a set of values for a working implementation and complement the literature's expectations in Section IV.

A. V2X QOS ANALYSIS

The measurement results provided by Qosium allow for deep analysis of empirical QoS for the V2X application and system. This paper's main results are split into two tables introducing delay and connectivity results.

1) DELAY RESULTS

The delay results are shown in Table 5. Our analysis considers the averaged result statistics using an averaging period of ($T = 1000ms$). Qosium calculates statistics for every packet and provides the results in the average form suitable for our study.

Right away, one can see three scenarios with high packet loss. The Campus scenarios with $150m$ and $250m$ have obstructions, such as trees and buildings, which lead to lost packets. The Autobahn scenario has out-of-range distances, which also hinders the reception of packets. The delay results are within the expected range, as shown in Figure 10. One highlight is that jitter, delay standard deviation, and maximum delay statistics values are about double higher in the city follower scenario than in the other scenarios. This is caused by differential speed and noise on the wireless channel, which produced lag spikes, as shown in Figure 11.

2) CONNECTIVITY RESULTS

The connectivity results are shown in Table 6. These results include the PLB lengths and their durations in time. A PLB spans from one lost packet until another is successfully received.

With the standstill vehicles, we measured only one of the 1719 packets lost during the measurement scenario. However, when considering vehicle mobility, we observe long PLBs. The Campus Round scenario with a $50m$ distance and the city follower example give important metrics for close city distances. The results show a $3-14\%$ packet loss ratio, accompanied by $29-86ms$ average connection loss, at worst $0.8-1.2s$. Even more, it is essential to note that obstructions impact packet loss substantially (see $150m/250m$ city scenarios). In such scenarios, packet loss causes longer latencies for the information itself. Also, the potential for lost messages gets more definite with longer distances, as seen in the Autobahn scenario. We tested it as an overtaking scenario, which leads to a high packet loss and long PLBs when going out of range.

For the delay components, shown in Table 1, three different transmission values visualize packet loss in transmission. First, an average of $5ms$ are for the closest possible vehicles.

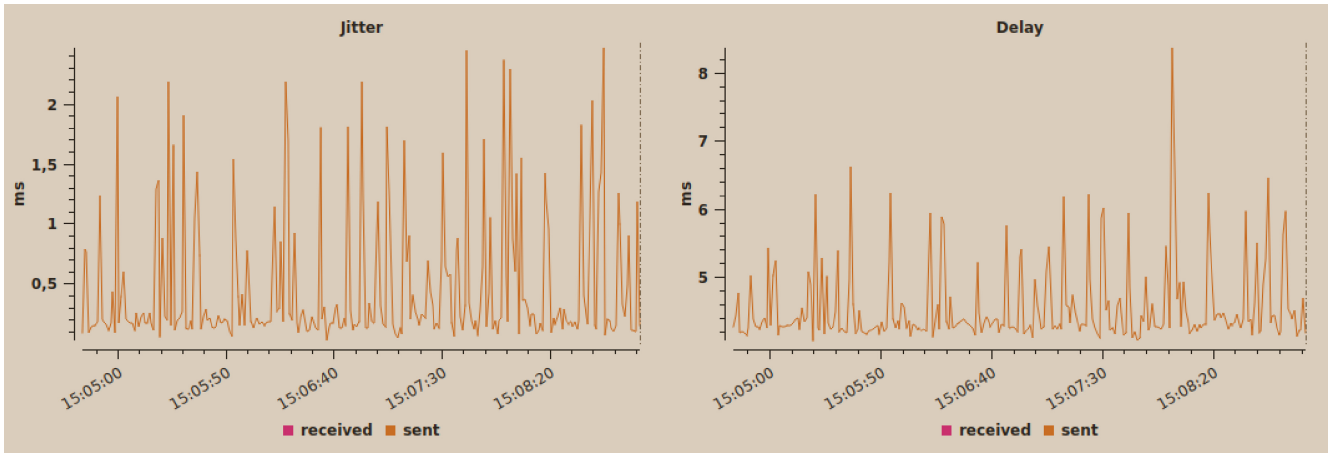

FIGURE 10. V2X QoS results for the Campus Round 50m scenario, with one sending ITS-S. The overall E2E transmission delay is between $4-8ms$. Most results are between $4-6ms$, confirmed by the jitter, which is between $0-2.5ms$. Both diagrams show a strong baseline, confirmed by the standard deviation of 0.58 from the averaging results shown in Table 5.

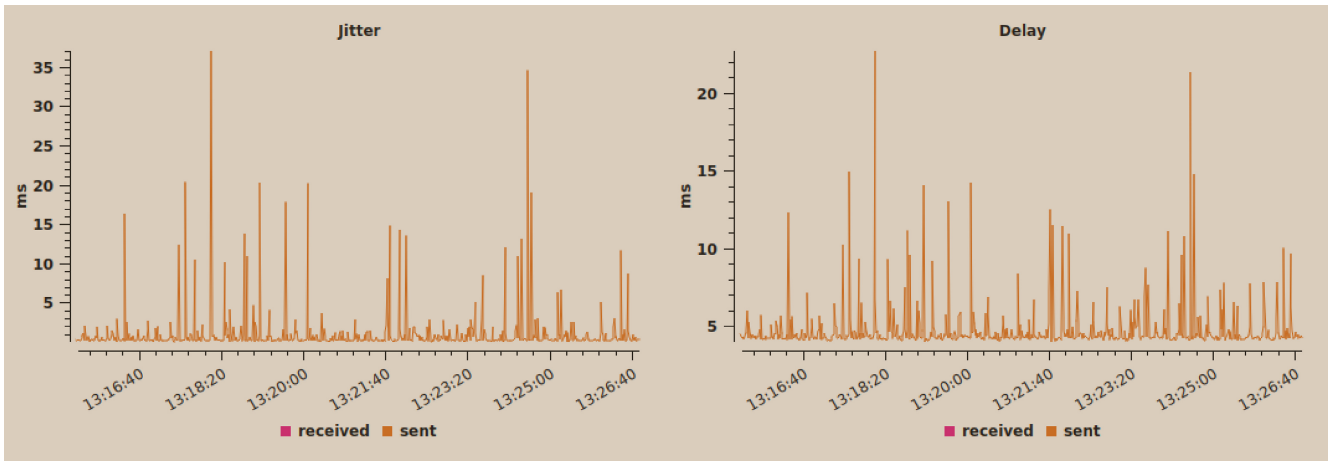

FIGURE 11. V2X QoS results for the City Follower Scenario, with one sending ITS-S. The E2E transmission delay is between $4-23ms$ in the averaging intervals. Most delays are between $4-10ms$, confirmed by the jitter of $0-37ms$, with most results between $0-5ms$.

This time should be used for calculations with vehicles in the line of sight without interference. The second one is for the average city scenario, where vehicles are within the theoretical range of onboard sensors but may already provide a view around the corner. This will increase the field of view (FOV) for both vehicles. In this case, a blocked line of sight is possible when one vehicle drives around the corner. Interference, fading, and signal strength contribute to lost messages. As can be seen in the measurement results, 3.71% of messages were lost in the city follower scenario. However, lost packets proliferate with tight corners and blocking objects, such as in the Campus scenario, 13.69% at $50m$. Similarly, the number of PLBs increases. The average PLB durations of $29ms$ and $86ms$ with a high standard deviation show that the communication becomes more unreliable than in the first scenario. The results indicate that most PLBs are due to losing only one packet, which should be considered in application development. Therefore, $105ms$ estimates one missed message, including the transmission time of an additional packet.

The third delay value is for longer distances. In most scenarios, there is a time when vehicles are close enough to each other to overcome the interference of obstructions, i.e., line of sight is mostly no problem. Issues arise when driving around the corner. With constant speed, we found that PLBs are, on average, $318-411ms$, i.e., about $3-4$ messages are lost. We, therefore, expect $405ms$, which covers four lost messages and one succeeding transmission. Higher delays are then in ranges where infrastructure-supported relaying and mobile network connections with server backends, i.e., V2N, may be more feasible.

3) CHANNEL LOAD

Channel load and differential speed have a major influence on the transmission delay, as discussed in Section IV-B3. However, the given setup is not able to create controlled artificial channel load or controlled differential speed. Hence one has to add expectations from simulations [28], [29]. For low channel loads the previously discussed measurements of $5/105/405ms$ are applicable, depending on the scenario.

TABLE 7. ROS topic frequencies - separate runs in Ford Fusion (*) and Ford Mondeo (+). Runs show node frequency stability.

ROS Topic	f	Δf_{min}	Δf_{max}	$\hat{\sigma}_f^2$	$samples$
camera A (*)	15.95	0.02	0.38	0.02	2803
camera B (*)	15.94	0.03	0.39	0.03	2803
lidar (*)	10.0	0.07	0.25	0.01	2878
CPM-send (*)	10.1	0.09	0.13	0.03	2878
CPM-recv (+)	10.0	0.05	0.07	0.01	3512

For medium channel load, $100 - 500ms$ have to be added. For high channel load, one has to estimate an additional $500 - \infty ms$ of delay, meaning delivery is not guaranteed, depending on channel load and message priority. For the final expectations in Table 1, we expect the three categories to also reflect the combination with differential speed, as rough major categories. A higher level of detail is provided in Section VII.

B. CP DELAY CHAIN ANALYSIS

The following paragraphs will fill the measurement column of the Components of Delay, shown in Table 1. The measurements are taken from three test drives around the Inffeld Campus of TU Graz. In Section V, we mentioned that our ADD runs Autoware, simplified as shown in Figure 5. The runtime of the nodes, shown in Table 8, is the time difference between the availability of an input message and the availability of the corresponding output messages. Keep in mind that also ROS2 communication is in between the components with around $1ms$ delay for each communication step [17]. Aside from these results, one must remember that some nodes' runtime depends on the number of detected objects.

1) SENSOR DATA DELAY

The sensor data delay is calculated as shown in Equation (2). The acquisition time can be taken from the datasheet, if available. For the Ouster OS1 Lidar [10] it is $< 10ms$. For the FLIR BFS-PGE-27S5C-C [11] camera, we have to estimate $23.3ms$ for $43FPS$ as upper data latency per frame as no concrete values are mentioned. To get a more accurate reading, one would have to set up a measurement setup where movement is created, and the output is measured on the other end. The second value for the data delay is the readout delay. The respective ROS2 nodes in our system need around $0.8ms$ for camera data and around $24.3ms$ for lidar data, as shown in Table 8. The resulting data delay is now $24.1ms$ for camera data and $34.3ms$ for lidar data.

2) SENSOR CYCLE TIME

The sensor cycle time is given by the frequency of the respective ROS2 nodes, which are $16Hz \Rightarrow 62.5ms$ for the camera and $10Hz \Rightarrow 100ms$ for lidar, as shown in Table 7.

3) OBJECT DETECTION DELAY

The object detection delay consists of the pre-processing and object detection steps. Pre-processing is necessary to rectify the camera image and decrease the point cloud's level of detail. In our case, the camera images are analyzed with the YOLO3 network, which results in 2D labeled object lists. For the lidar, we get unlabeled 3D bounding boxes from a convolutional neural network (CNN). Overall, the pre-processing and object detection takes around $69.0ms$ for camera data and $73.4ms$ for lidar data, as shown in Table 8. Finally, as discussed in Section IV, one should know that data of similar sensors are fused before the object detection and pre-processing step. For example, our ADD Ford Fusion can fuse the sensor images of four separate Ouster lidars. However, for simplicity, we only use one Ouster lidar for the tests conducted in this paper. Hence we get the final sensing delays as $159.8ms$ for camera data and $207.7ms$ for lidar data with Equation (4). For Table 8, we use the higher delay for lidar, which is within the expected range.

4) DATA FUSION (LOCAL)

The local fusion is done at the object level when the camera and lidar object lists are ready. The fusion is done in image space. Therefore, 3D objects are first projected to the image space and compared with the object list of the YOLO3. The objects are then merged based on the intersection of the unions. In an additional step, 3D shape fitting is done based on the classification. This fusion takes around $0.3ms$ to process object data from the camera and lidar in our system. However, it may be higher if many new objects are detected, as the maximum of $11.12ms$ shows. As data fusion is a single processing node, the ROS2 DDS delay of $< 1ms$ [17] is added, resulting in $1.3ms$ data fusion time.

5) MATCHING & TRACKING (LOCAL)

Matching, also called data association, is done right before the tracking stage. The tracked objects from the previous round are matched with the new objects of the fusion step with a global nearest neighbor approach. Tracking is the final step in the local perception pipeline. The matched objects are now used to update the tracked objects from the previous round or, in case no match was found, to initialize new objects for tracking. Matching and tracking take around $0.1ms$. Both are done in a single ROS2 node. Hence one has to add once the ROS2 DDS delay of $< 1ms$ [17], which results in $1.1ms$ for the matching and tracking and $2.4ms$ for the overall fusion.

6) BEYOND THE PERCEPTION PIPELINE

After the tracking, the local perception is complete. Next, our system will predict the object's movement from the existing track. The local perception and the predict-ahead are then used by the planner, which then acts upon the data.

7) BUFFERING

The local perception can be buffered with a sensor-like frequency of 10Hz. In our implementation, this frequency is already provided as our local fusion output, which is

TABLE 8. Delay test results for each CP component from the ADD Ford Fusion. Summary of three test rounds around the Inffeld Campus of the Graz University of Technology.

CP Component	$\sum$	$\tau_{min}[ms]$	$\tau_{max}[ms]$	$\overline{\tau}[ms]$	$\widetilde{\tau}[ms]$	$\hat{\sigma}^2_\tau[ms]$
Data Latency						
|- Camera A	9743	0.480	11.163	0.807	0.734	0.256
|- Camera B	8543	0.303	4.167	0.555	0.481	0.217
|- Lidar	9746	15.983	99.148	24.255	20.283	7.734
Object Detection						
|- Camera	5696	3.241	72.250	69.018	69.261	3.179
|- Lidar	5688	56.966	282.586	73.416	71.319	11.897
Fusion	5582	0.007	11.132	0.266	0.242	0.213
Tracking	5582	0.004	0.424	0.074	0.062	0.045
Packing	10006	0.000	0.250	0.103	0.117	0.038
Unpacking	10006	0.000	0.102	0.030	0.034	0.012

stable 10Hz, as shown in Table 8. In future implementations, we will add filters to prevent congestion, as suggested by ETSI specifications [1]. For now, there is only the ROS2 DDS delay that routes the data through an empty ROS2 node. Hence $< 1ms$ ROS2 DDS delay [17].

8) PACKING

The packing is done in our V2X gateway. As soon as the CPM node sends out the ROS2 CPM message, the V2X gateway converts the ROS2 message into an ETSI conform ASN1 message and sends it to the vehicleCAPTAIN. This process takes around $0.103ms$ for a CAM. CAMs are used for reference, as CPMs are not yet fully standardized and can not easily be compared. Afterward, as discussed during QoS measurements, the vehicleCAPTAIN transmits the ASN1 payload. For such small times, we add the ROS2 DDS delay with $< 1ms$ [17], resulting in $1.103ms$.

9) UNPACKING

The unpacking is similar to packing. The V2X gateway polls the vehicleCAPTAIN with a maximum sleep time of $10ms$ if messages are not received in between. The data is collected from the vehicleCAPTAIN and converted from the ASN1 bitstream to a ROS2 CPM. The conversion takes around $0.030ms$ for a CAM. CAMs are used for comparison, as stated above, for packing. Including the $< 1ms$ ROS2 DDS delay [17], this is up to $11.030ms$ of unpacking time.

10) DISTRIBUTION

The CPM node of the receiver distributes CPMs as sensor messages. In our implementation, we directly feed the incoming data into the global fusion pipeline, which results in a $< 1ms$ ROS2 DDS delay [17]. In a future implementation, we aim to pre-process V2X data to decrease the computational load of multiple V2X messages on the global fusion.

11) ASYNCHRONOUS FUSION DELAY

If one distributes V2X messages with a specific frequency, the global fusion could be run directly afterward. In our implementation, we directly relay V2X messages to ROS2 so that other systems can immediately use them. As one fusion step takes around $2.4ms$, the fusion could run at $100Hz$, which would add $10ms$ of worst-case delay. However, this high frequency will not be possible with multiple CPM sources, as discussed in Section VIII. As the local perception pipeline runs with $10Hz$, we also limit the global fusion to $10Hz$. This adds a worst-case delay of $100ms$ if the fusion cycle is missed.

12) FUSION, MATCHING, TRACKING (GLOBAL)

Global fusion components will differ from local fusion components depending on the fusion strategy. The problem is how to fuse multiple CP sources. In our measurement, with only one CP source, the fusion delay is the same as in the local fusion, i.e., $2.4ms$ for two ROS2 nodes and their processing time. However, fusing each CP source as a separate sensor will lead to scaling problems. Five or ten CP sources will not be a problem, as the situation is similar to a multi-sensor setup in automated vehicles. But especially when V2N sources are integrated, there has to be a fusion strategy, which increases the overall fusion time.

VII. THEORY OF TRANSMISSION DELAY WITH ITSG5

This section goes into more technical detail on ITSG5-based V2X communications to better understand where most of the communications delay comes from. As discussed in Sections IV-B3) and VI-A3), loaded channels can result in high delays. Depending on several factors – such as the type of messages (priority) and transmission (unicast and broadcast), message size, signal strength, channel noise and interference, antenna setups, signal propagation factors, and the number of actively transmitting ITS-Ss – a message may experience increased delay, affecting the buffered messages sent next, or a failed transmission.

One major challenge with ITSG5 is multiuser scenarios, which affect delays and reliability. Only one station can send at a time over a single channel while the others need to wait for their turn. The Medium Access Control (MAC) layer in ITSG5 is based on Enhanced Distributed Coordination

Access (EDCA) which, similarly to Distributed Coordination Function (DCF), gives every station the power to compete for radio resources [27]. EDCA brings additional capabilities over DCF to prioritize different types of traffic by introducing data traffic-specific queues and different listening periods called Arbitration Interframe Space (AIFS). The AIFS length depends on the access category; where AIFS is shorter, the higher the priority. This results in a situation where channel access for low-priority traffic can be substantially delayed; the lowest-priority traffic can even experience starvation with much high-priority traffic [29], [37], [38], [39]. Traffic with short AIFS can proceed to direct transmission with a much higher probability than low-priority packets.

ITSG5 employs Carrier-Sense Multiple Access (CSMA) with Collision Avoidance (CA). If the channel is busy upon transmission attempt, the station goes to a backoff state. The key to avoiding different transmitting stations sending simultaneously is the random duration of the backoff period, selected from the range [0, CW]. Thus, the Contention Window (CW), with defined CWmin and CWmax values for different access categories, plays an essential role in the backoff duration. The backoff counter is decremented only when the channel is idle. On busy channel moments, the stations in backoff listen to the AIFS period, specific to the access category, until the channel becomes free again to continue decreasing the backoff counter. Thus, the CW size and the AIFS duration impact the time to wait for transmission turns. Upon collision or otherwise unsuccessful transmission, the CW is increased until CWmax and retransmission limit is reached, potentially extending the next backoff period for the retransmission. This is to spread transmissions from multiple stations in time, but it potentially introduces additional delays, especially for background and best-effort traffic categories. The retransmission applies only to the unicast mode, while in broadcast, there is only one backoff period, after which the packet is sent, successfully or not, to ITS-S receivers in range. Broadcast traffic does not suffer from increased delays caused by retransmissions, but, on the other hand, there is no information on whether all stations benefitting from the message got it. One challenge of the EDCA and DCF, also impacting the V2X scenarios, is the hidden node problem. If two transmitting stations cannot hear each other, they can initiate transmission at the same time, resulting in a collision.

ITSG5 is based on a random-access MAC. The number of competing stations, traffic access category, packet sizes, retransmissions, and packet buffer lengths impact the delay in ITSG5 systems on the MAC level, being spiced up by the physical layer (PHY) and applications. Our results provide an empirical evaluation of the V2X delays without ITS-Ss competing for the same radio resources. The CPM is of the highest priority, with only ten packets per second per ITS-S, and when broadcast, the delays, in theory, do not grow high. Comparing our results with related work, based on simulations, for accurately determining the practical delay behavior in congested channels is not straightforward. As no study that empirically evaluates the V2X communications delay in controlled channel loading scenarios with traffic of different access categories and multiple transmitting ITS-S exists, this is an interesting future work item, also to discover how the estimations from the literature of this paper match the practice.

VIII. DISCUSSION

The overall time needed for all CP components — from an event happening until reaching the awareness of it within another ITS-S — can be derived as discussed in Section IV. Table 1 shows the estimations from literature and includes the field operational performance metrics provided in Section VI. The discussion below will first analyze the influence of QoS parameters on the transmission, followed by analyzing essential parts of the CP chain, and finally conclude by looking at the applicability of CP in the scenarios specified in Section II.

A. INFLUENCE OF QOS

The V2X QoS measurements confirmed the low transmission times of V2X messages in side-by-side scenarios, as shown in Section VI. The low latency times of around $5ms$ are applicable for specialized time-critical scenarios, such as virtual towing [36], where the packet loss should be close to zero. However, if a vehicle's sensors are working correctly, the nearby surrounding of a vehicle will always be covered with onboard sensors. Hence the following form of CP scenarios is applicable for city scenarios, where distances are between $50 - 100m$. In these cases, we expect only to receive every other V2X message, as one message can be lost. Thus, $105ms$ is a reasonable estimation for a missed message and the delay of the next one. Finally, there are scenarios with farther distances of $150 - 250m$. In these scenarios, the packet loss is high, primarily due to occlusion. However, those scenarios are edge scenarios for CP. They mean that two vehicles are, e.g., occluded by a long corner on a highway or by buildings in a city. Other scenarios are already a city scenario with close distances and occlusions or a scenario with a line of sight, where V2X is working again. Hence, $405ms$ is selected for those ranges. For such scenarios, it is also essential to mention that fast V2N scenarios may be in similar ranges. Correia et al. [26] shows a $200 - 6000ms$ delay with LTE via a single MQTT server. However, their setup is straightforward, and delays with traffic management software would be more applicable. I.e., the V2N messages of vehicles have to be managed to be only relayed in specific regions, or else the mobile network is also congested.

However, beyond our field evaluation, one has to be aware of scalability issues resulting from loaded channels, as well as differential speed. As stated in Section VII and compared to literature in Section IV, the back-off algorithm and EDCA system can cause delay $>> 100ms$. To be specific, simulations by Shah et al. [28] suggest an additional delay of

TABLE 9. Local perception compared to expected extra delay caused by CP. Comparison of literature (lit) to field operational (real) by scenario. Premise: local perception can detect the object. Difference between CP and local perception Δt. All delays in [*ms*].

Scenario	$local_{lit}$	$local_{real}$	$\Delta t_{lit,low}$	$\Delta t_{lit,med}$*	$\Delta t_{lit,high}$*	$\Delta t_{real,low}$	$\Delta t_{real,mid}$*	$\Delta t_{real,high}$*
Side-by-Side	$208-282$	207.7	$15-47$	$115-547$	$515-\infty$	123.9	$223.9-623.9$	$623.9-\infty$
City	$208-282$	207.7	$115-147$	$215-647$	$615-\infty$	223.9	$323.9-723.9$	$723.9-\infty$
Highway	$208-282$	207.7	N/A	N/A	N/A	523.9	$623.9-1023.9$	$1023.9-\infty$

* Scalability: additional delays for medium/high channel load and differential speed, as discussed in Sections IV.B.3 and VI.A.3.

$100-500ms$ for medium channel load and differential speed, depending on the definition of medium load and differential speed. For high channel load, the delay is unbounded. In other words, a message may never arrive, depending on multiple factors, such as priority and involved ITS-Ss. Hence we suggest a discussion on including the V2X channel load into the global fusion algorithm. To be specific: if, e.g., a free space message is sent, its validity should depend on the channel load, as the negation of the free space may not arrive in time or at all.

One big remaining issue is the scheduling of V2X messages. If the CP algorithm [1] does not consider the CP object as critical, the sending frequency is only 1Hz, which increases the expected delay by a factor of 10. This delay will also defeat the purpose of thinking about lost consecutive messages, as four lost messages from the expectation above would lead to $4s$ of delay; this, in turn, raises the question of how the situation evolved four seconds later. One could argue that the object is not critical in this case, else another 10Hz burst would have been sent, which in turn throws overboard all estimations on the frequency and expected lost messages. In conclusion, for the discussion in this paper, we expect the object to be critical.

B. CP DELAY CHAIN

It is essential to be aware of the entire CP delay chain to discuss the implications of changes in specific components. This paper overviews the CP delay components in Figure 4. To get the delay overhead of CP, one can subtract the local perception delay from the overall delay. Table 9 shows the expectations from the literature compared to an analysis of the implementation in this paper. The delta time is the delta between local perception and total CP delay. If local sensing and fusion are similar for both vehicles, the difference in time is primarily the added communication delay and the global fusion delay. However, as seen in Table 1, there is also a delay for asynchronicity. Literature shows this delay within the communication part. But as our implementation showed, this asynchronous delay can also be in the global fusion.

For the discussion, there are two important delays. The first one is the age of information. This is the delta, which is caused by transmission and global fusion. We show that CP information takes at least 1.5 times longer to be available than local perception, as shown in Table 1. The other important delay concerns the first information about an event. This delay takes packet loss into account. The QoS analysis of ITSG5 confirmed that one should expect one missed package in city scenarios, which means that global perception is twice as slow as local perception. Higher distances lead to weaker connectivity and more packet loss. Hence, higher delay of information, meaning the global perception is > 2.5 times the local perception.

If the channel is loaded, or the differential speed is high, the delay of information increases drastically, as discussed in Section IV. The tests in this work are limited by not having the capability for controlled channel loading and controlled differential speed. Hence Table 9 shows expected delays for medium and high channel load, respectively differential speed, with added estimations from simulations by Shah et al. [28].

C. SCENARIOS

With the complete CP delay chain, one can now discuss the applicability of CP concerning safety and comfort. At this point, one must emphasize that state-of-the-art automated vehicles always have to carry the consequences for their actions, similar to a human driver. Thus, an automated vehicle has to drive in a way such that no critical situations arise. In general, a human driver learns in Austria/Germany that a child may walk out between vehicles, and one has to be able to stop in advance. By current standards, this is also true for an automated vehicle. The automated vehicle has to be able to stop in advance. In other words, CP may not be used to prevent such situations, as the automated vehicle has to guarantee that. However, evaluating the delay contributes to discussing whether the general approach could change.

In general, CP may be used to improve traffic flow and comfort. If the ego vehicle knows that objects (VRUs, other vehicles, debris, or similar obstacles) do not occupy a particular area, it may increase its speed. In these cases, it is crucial if the situation changes. In other words, if an RSU sends out free-space messages via CP and the situation changes, countermeasures must be fast. There are three types of countermeasures: (i) the warning via V2X, which is the fastest if there is no direct knowledge about a situation, (ii) the onboard detection, which is the standard for an automated vehicle and (iii) the human intervention, which is comparatively slower to the onboard detection, as elaborated in the following.

Table 9 shows that CP data is $123.9-523.9ms$ slower than onboard sensor data within low channel load scenarios. Thus, as long as the ego vehicle can not detect the dangerous object within this time difference, V2X is faster.

However, C-ITS will suffer from loaded channels, with a higher density of ITS-Ss. In city scenarios, this is likely to cause $100 - 500ms$ of additional delay. Depending on the circumstances, the delay may be drastically higher, up to a point where the message can not successfully be sent. Beyond transmission delays, the implementation of the tested CP chain does not include signed data or any other form of verification. Thus, the data may not be trusted, which is terrible for safety-relevant situations, as counter-actions may lead to worse situations. Therefore, verification and trust management delays have to be considered but have yet to be measured. Security delays aside, if CP information extends the FoV, the scenarios discussed in Section II are applicable.

Last but not least, there is human intervention. The human reaction time for objects moving into the drive path is roughly 1.5 seconds, according to Green [40]. As Table 1 shows, the sensing time is up to $280ms$ as suggested by literature and $207.7ms$ tested with our vehicle. Thus, an automated vehicle will react faster and more precisely to date. In contrast, a human may be better in their decision if it is a trained driver or in uncertain situations. However, an automated vehicle can be more aware of its surroundings, which may be even better with V2X.

The following paragraphs will discuss the three reaction methods to better understand how V2X can contribute to safety and comfort. All scenarios present lower expectations without channel load. The transmission delay is basically unbounded, but at least $>> 100ms$ for loaded channels, as discussed in Section IV.

1) CITY SIDE-BY-SIDE SCENARIO

The city scenario, shown in Figure 1, has a VRU that will come out between vehicles. As discussed above, an automated vehicle has to be aware of the possibility of such a situation. In that sense, the speed has to be according to the situation. Thus, safety should be no problem. However, it can increase comfort and performance. If the oncoming ego vehicle is aware of a declared free space or a VRU between the vehicles, it can act accordingly. With free space declared, the ego vehicle can increase its speed. The sensors of the free space declaring ITS-S will send out a CP warning of the pedestrian, which will be taken into account by the ego vehicle $123.9ms$ later, due to the close distance, without channel load. If a VRU approaches the free space, the situation changes again, triggering a CP. Hence there will not be the possibility of a sudden pedestrian appearance. Hence the ego vehicle has to adapt its speed according to the available free space that can cover predict-ahead estimations of VRUs, including the expected CP delay. The scenario is similar if there is already a VRU between the vehicles. In conclusion, if the CP data of the VRU can be integrated before the data of the onboard sensors, the ego vehicle will have a time advantage due to the CP data.

2) HIGHWAY SCENARIO

The highway scenario shown in Figure 2 has a broken-down vehicle with a VRU walking around. As discussed above, such a situation should be no safety issue. However, the ego vehicle can increase speed if an ITS-S declares free space. In this case, the ego vehicle has to consider the delay caused by lost packages at long distances. In other words, if the situation changes for a particular area, at least $405ms$ has to be added for the transmission time. This factor is especially true when there is no RSU but oncoming vehicles that declare the free space. In this case, there might always be a situation where oncoming vehicles have sensor range to the location of the scenario, but the ego vehicle is on maximum communication range. However, one must consider the arrival of early information that can neither be known by the ego vehicle nor by an active human driver. In that case, at least $523.9ms$ of extra delay is better than having no information until getting into the sensor range. Looking further, if there is no free space but an obstacle, as shown in Figure 2, the ego vehicle can ensure the comfort of its passengers and slow down in advance, i.e., avoiding driving on the limit.

3) CITY INTERSECTION SCENARIO

The intersection scenario shown in Figure 3 has an intersection scenario where a VRU crosses the road behind the corner. Again, safety should not be an issue here, as the ego vehicle has to drive in a way where onboard sensors can avoid a collision. Overall, the scenario is similar to the city scenario; however, messages could be lost due to corners. In this case, we consider that an ITS-S may send out free space around the corner, but the situation changes. As the situation changes, the ITS-S detecting it immediately ($10Hz$) sends out the position of a VRU crossing the street. Nevertheless, as it is a city scenario, one message may be lost. This results in at least $223.9ms$ additional delay, compared to the onboard perception. Depending on the slow speeds at the intersection, this may be fast enough, depending on the channel load.

IX. RELATED RESEARCH

This section points out related research, which was not directly looked at in this paper, but influences the usage of V2X in general or CP in specific.

One parameter not directly looked at in this work is channel load and congestion. If the V2X channel is congested, a CPM is not guaranteed to arrive, as it could be overridden by multiple high-priority messages, such as DENMs and CAMs. However, preventing congestion caused by CPMs is well-studied. Many studies deal with data selection to decrease the size of CPMs. Methods are to mitigate redundancy [41], pack detected objects into one instead of multiple messages [15], [16], and decide which objects might be necessary at all [42], [43], [44]. These methods decrease the channel load, which is essential not to increase the channel-busy ratio.

Another evolving parameter is the discussion around security, especially the trustworthiness of CP data. Related

research deals with V2X cyber security challenges [45], [46], [47], [48], but also the need to reflect trustworthiness in the data [49], [50], [51], [52]. Depending on the methods used, delay times for CP may increase. The systematic approach to evaluating the CP delay chain presented in this paper allows a discussion on the influence of this delay.

We also did not directly address different forms of fusion algorithms, i.e., the implementation of particle filters and occupancy grids, as discussed by Godoy et al. [53]. The planner of Autoware works with a map. This map can either be generated with perception from the onboard sensors or preinstalled. For this paper, we tested the perception stack, as discussed above, as the planning algorithm is unnecessary for the CP delay evaluation.

We focused on cmWave protocols for data transmission, with ITSG5 as an example. The lower modulation of the transmission frequency results in a limited bandwidth, which only allows the transmission of object data, as specified by ETSI [1]. However, there is also the evolving mmWave standard with ten times higher frequency in the 60GHz area. This frequency allows the transmission of the raw camera and lidar data, as discussed in the related literature [54], [55], [56]. As discussed in previous research [3], using raw data for transmission will also influence the fusion. It will be interesting to see how the CP delay chain must be adapted when this technology is ready to be integrated.

Finally, we did not dig deeper into the domain of controller theory. As discussed, our implementation uses Autoware as the automated driving stack. Autoware is a collection of best practices to balance the overall system's performance. In that regard, analyzing Autoware from a control perspective would be interesting. Platooning can contribute much to the control analysis by analyzing the control loop in connected vehicles [57], [58], [59], [60], of course also dealing with unknown communication-related delays [61], [62], or fixed expected frequencies, [63].

X. CONCLUSION

This paper aims to provide a systematic view of the delays of CP. We started by defining a set of scenarios where CP can provide benefits. We then defined the delay elements of CP and collected expectations from the literature for all elements. We then set up two measurement campaigns (i) to deepen the knowledge of V2X QoS performance with highly time-accurate measurements in low channel load scenarios and (ii) to complete the delay dataset of the CP delay chain with field operational performance measurements, of a V2X enhanced automated driving stack.

The update of the CP delay chain allowed us to collect delay expectations from the literature. We then used a state-of-the-art Autoware implementation in our street-legal ADDs to collect delay results in field operational tests. We first confirmed that low latency can be fulfilled for CP with field operational QoS parameters of ITS-G5 with around $5ms$ transmission time at close distances in sparse traffic scenarios. We then argued that packet loss should be counted as additional $10Hz$ cycle delays for high-priority CPMs, leading to an expected information delay of $105ms$ for intersection scenarios in cities and $405ms$ for highway scenarios.

The results are limited by not specifically testing loaded or congested communication. Hence, we highlight the risk in scalability, with transmission delays $>> 100ms$ due to necessary MAC mechanisms. To be more specific, we derived an overhead of an additional $100 - 500ms$ delay for channels with medium load and $35 - 120km/h$ differential speed and highlighted that highly loaded channels have unbounded delays. Consequently, we started the discussion that loaded channels should be part of the trustworthiness of CP data. In other words, if the channel is loaded, CP data will have higher delays, which results in risks, e.g., CP updates negating free space messages may never arrive.

The discussion of delays of the full CP chain also showed that a basic high delay originates in the asynchronous reception cycle. Either the CP data is received and distributed with a specific frequency before feeding it into the global fusion, or the global fusion cycle creates an asynchronous delay with its cycle. This results in a $100ms$ cycle delay caused by the $10Hz$ global fusion for state-of-the-art automated driving software, as demonstrated in this work. Loaded channels will produce different results, making the transmission delay more prominent. However, the provided delay expectations from the literature and a state-of-the-art system provide a basis for the optimization discussion.

Finally, the expected local perception delay — for sparse traffic scenarios, as defined in Section II — is $207.7ms$. We argue that close-distance scenarios are a minority of use cases, as the onboard sensors should be able to cover the driving path. Hence, while the expected additional global perception delay is $123.9ms$, the expected additional information delay is at least $223.9ms$, due to packet loss. In other words, in low-traffic scenarios, CP information should be expected to have at least twice the delay of local perception. However, the delay may increase drastically in higher V2X channel load cases.

As a follow-up for this work, we see (i) the integration of V2N for CP and (ii) the extension of tests for multiple global fusion sources. The usage of V2N with 5G benefits from the low latency of 5G systems, as shown by Castiglione et al. [64]. However, the integration of V2N will increase the overhead with a backend system, as shown by Correia et al. [26]. It will also need a management system for data exchange dedicated to geo-location. The advantage is the higher reliability for guaranteed data delivery. The disadvantage is the much higher delay with two-way connection and backend management. The V2N connection will also create challenges for global fusion. As discussed in this work, our global fusion approach can handle limited CP sources. More than a dozen CP sources will be problematic, as the global fusion strategy has to manage the fusion of multiple input sources with various time delays. However, we see much potential in applying existing fusion strategies for onboard sensors in robotics. But we also see difficulties

with the additional time delay caused by loaded channels, which might greatly negate the need to optimize the global fusion approach.

Beyond this work, one should consider the implications of CP for safety purposes. Three immediate issues are: (i) first, lifting safety guards for higher performance can lead to dangerous situations. Not only when the CP chain is broken at some point, i.e., packet loss or unbound MAC delay, but mainly when unsecured towards adversaries and faults. Parameters on when to allow the lifting of safety guards have to be precise; (ii) second, increasing comfort by braking in advance to prevent emergency braking for VRUs can lead to psychological issues with legacy vehicles. In other words, future VRUs will learn that vehicles brake even though they can not see them, so they are safe. If the oncoming vehicle is now a legacy vehicle, without CP functionality, fatal accidents may happen; and (iii) third, ignoring hard rules, such as stopping at red traffic lights, stopping at stop signs, ignoring speed limits, and many more, may create uncertainty in hybrid environments.

Finally, CP can increase safety and comfort by extending the FOV as long as onboard safety can be guaranteed. Beyond that, other factors, such as reliability and accuracy, play an essential role.

ACKNOWLEDGMENT

InSecTT (www.insectt.eu) has received funding from the ECSEL Joint Undertaking (JU) under grant agreement No 876038. The JU receives support from the European Union's Horizon 2020 research and innovation programme and Austria, Sweden, Spain, Italy, France, Portugal, Ireland, Finland, Slovenia, Poland, The Netherlands, and Turkey. The document reflects only the author's view and the Commission is not responsible for any use that may be made of the information it contains. The publication was written at Virtual Vehicle Research GmbH in Graz and partially funded by the COMET K2 Competence Centers for Excellent Technologies from the Austrian Federal Ministry for Climate Action (BMK), the Austrian Federal Ministry for Digital and Economic Affairs (BMDW), the Province of Styria (Dept. 12) and the Styrian Business Promotion Agency (SFG). The Austrian Research Promotion Agency (FFG) has been authorised for the programme management.

REFERENCES

[1] "Intelligent transport system (ITS); vehicular communications; basic set of applications; specification of the collective perception service, Release2," ETSI, Sophia Antipolis, France, ETSI TS 103 324 V0.0.52, Dec. 2022.

[2] A. Rauch, F. Klanner, and K. Dietmayer, "Analysis of V2X communication parameters for the development of a fusion architecture for cooperative perception systems," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2011, pp. 685–690.

[3] C. Pilz, A. Ulbel, and G. Steinbauer-Wagner, "The components of cooperative perception—A proposal for future works," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, 2021, pp. 7–14.

[4] F. A. Schiegg, I. Llatser, D. Bischoff, and G. Volk, "Collective perception: A safety perspective," *Sensors*, vol. 21, no. 1, p. 159, 2021. [Online]. Available: https://www.mdpi.com/1424--8220/21/1/159

[5] G. Volk, Q. Delooz, F. A. Schiegg, A. Von Bernuth, A. Festag, and O. Bringmann, "Towards realistic evaluation of collective perception for connected and automated driving," in *Proc. IEEE Int. Intell. Transp. Syst. Conf. (ITSC)*, 2021, pp. 1049–1056.

[6] G. Cui, W. Zhang, Y. Xiao, L. Yao, and Z. Fang, "Cooperative perception technology of autonomous driving in the internet of vehicles environment: A review," *Sensors*, vol. 22, no. 15, p. 5535, 2022. [Online]. Available: https://www.mdpi.com/1424--8220/22/15/5535

[7] A. Böhm, K. Lidström, M. Jonsson, and T. Larsson, "Evaluating CALM M5-based vehicle-to-vehicle communication in various road settings through field trials," in *Proc. IEEE Local Comput. Netw. Conf.*, 2010, pp. 613–620.

[8] B. Altinel, C. Bornkessel, and M. A. Hein, "Wave-physical factors determining the link quality in ITS-G5 studied with field-operational tests," in *Proc. 15th Eur. Conf. Antennas Propag. (EuCAP)*, 2021, pp. 1–5.

[9] T. Photometrics.s "Readout vs. frame rate." 2022. [Online]. Available: https://www.photometrics.com/learn/imaging-topics/readout-vs-frame-rate

[10] I. Ouster. "Ouster OS1 datasheet." 2021. [Online]. Available: https://data.ouster.io/downloads/datasheets/datasheet-rev06--v2p2-OS1.pdf

[11] T. F. LLC. "FLIR blackfly S gigE." 2022. [Online]. Available: https://www.FLIR.com/products/blackfly-s-gige/

[12] J. Hackett and M. Shah, "Multi-sensor fusion: A perspective," in *Proc. IEEE Int. Conf. Robot. Autom.*, vol. 2, 1990, pp. 1324–1330.

[13] L.-H. Wen and K.-H. Jo, "Fast and accurate 3D object detection for Lidar-camera-based autonomous vehicles using one shared Voxel-based backbone," *IEEE Access*, vol. 9, pp. 22080–22089, 2021.

[14] G. Thandavarayan, M. Sepulcre, and J. Gozalvez, "Redundancy mitigation in cooperative perception for connected and automated vehicles," in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, 2020, pp. 1–5.

[15] G. Thandavarayan, M. Sepulcre, and J. Gozalvez, "Generation of cooperative perception messages for connected and automated vehicles," *IEEE Trans. Veh. Technol.*, vol. 69, no. 12, pp. 16336–16341, Dec. 2020.

[16] Q. Delooz and A. Festag, "Network load adaptation for collective perception in V2X communications," in *Proc. IEEE Int. Conf. Connect. Veh. Expo (ICCVE)*, 2019, pp. 1–6.

[17] T. Kronauer, J. Pohlmann, M. Matthé, T. Smejkal, and G. Fettweis, "Latency analysis of ROS2 multi-node systems," in *Proc. IEEE Int. Conf. Multisens. Fusion Integr. Intell. Syst. (MFI)*, 2021, pp. 1–7.

[18] W. Anwar, N. Franchi, and G. Fettweis, "Physical layer evaluation of V2X communications technologies: 5G NR-V2X, LTE-V2X, IEEE 802.11bd, and IEEE 802.11p," in *Proc. IEEE 90th Veh. Technol. Conf. (VTC-Fall)*, 2019, pp. 1–7.

[19] C. Campolo, A. Molinaro, A. O. Berthet, and A. Vinel, "On latency and reliability of road hazard warnings over the cellular V2X Sidelink interface," *IEEE Commun. Lett.*, vol. 23, no. 11, pp. 2135–2138, Nov. 2019.

[20] Z. Amjad, A. Sikora, B. Hilt, and J.-P. Lauffenburger, "Low latency V2X applications and network requirements: Performance evaluation," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2018, pp. 220–225.

[21] H.-J. Günther, R. Riebl, L. Wolf, and C. Facchi, "Collective perception and decentralized congestion control in vehicular ad-hoc networks," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, 2016, pp. 1–8.

[22] E. Xhoxhi and F. Schiegg, "Benefits of DCC facilities in ITS-G5 networks - first simulated results," in *Proc. IEEE 95th Veh. Technol. Conf. (VTC-Spring)*, 2022, pp. 1–7.

[23] T.-K. Lee, J.-J. Chen, Y.-C. Tseng, and C.-K. Lin, "Effect of packet loss and delay on V2X data fusion," in *Proc. 21st Asia-Pac. Netw. Operat. Manag. Symp. (APNOMS)*, 2020, pp. 302–305.

[24] G. Xiog, T. Yang, M. Li, Y. Zhang, W. Song, and J. Gong, "A novel V2X-based pedestrian collision avoidance system and the effects analysis of communication delay and packet loss on its application," in *Proc. IEEE Int. Conf. Veh. Electron. Saf. (ICVES)*, 2018, pp. 1–6.

[25] Z. Wang, "Motion estimation of connected and automated vehicles under communication delay and packet loss of V2X communications," SAE WCX Digital Summit, SAE Int. Warrendale, PA, USA, Apr. 2021. [Online]. Available: https://doi.org/10.4271/2021--01-0107

[26] M. Correia, J. Almeida, P. C. Bartolomeu, J. A. Fonseca, and J. Ferreira, "Performance assessment of collective perception service supported by the roadside infrastructure," *Electronics*, vol. 11, no. 3, p. 347, 2022. [Online]. Available: https://www.mdpi.com/2079--9292/11/3/347

[27] "Intelligent transport systems (ITS); ITS-G5 access layer specification for intelligent transport systems operating in the 5 GHz frequency band," ETSI, Sophia Antipolis, France, ETSI EN 302 663 V1.3.1, Jan. 2020.

[28] A. F. M. S. Shah, H. Ilhan, and U. Tureli, "Modeling and performance analysis of the IEEE 802.11P MAC for VANETs," in *Proc. 42nd Int. Conf. Telecommun. Signal Process. (TSP)*, 2019, pp. 393–396.

[29] J. Zheng and Q. Wu, "Performance modeling and analysis of the IEEE 802.11p EDCA mechanism for VANET," *IEEE Trans. Veh. Technol.*, vol. 65, no. 4, pp. 2673–2687, Apr. 2016.

[30] A. Foundation. "Autoware foundation homepage." 2021. [Online]. Available: https://www.autoware.org/

[31] B. S. Jahromi, T. Tulabandhula, and S. Cetin, "Real-time hybrid multi-sensor fusion framework for perception in autonomous vehicles," *Sensors*, vol. 19, no. 20, p. 4357, 2019. [Online]. Available: https://www.mdpi.com/1424--8220/19/20/4357

[32] A.-J. Gallego, J. Calvo-Zaragoza, and J. R. Rico-Juan, "Insights into efficient k-nearest Neighbor classification with convolutional neural codes," *IEEE Access*, vol. 8, pp. 99312–99326, 2020.

[33] V. V. R. GmbH. "Vehicle communication platform to anything–toolbox." 2023. Accessed: Feb. 13, 2023. [Online]. Available: https://github.com/virtual-vehicle/vehicle_captain

[34] T. Blazek, G. Ghiaasi, C. Backfrieder, G. Ostermayer, and C. F. Mecklenbräuker, "IEEE 802.11p performance for vehicle-to-anything connectivity in urban interference channels," in *Proc. 12th Eur. Conf. Antennas Propag. (EuCAP 2018)*, 2018, pp. 1–5.

[35] C. Belagal Math, H. Li, S. Heemstra de Groot, and I. G. Niemegeers, "V2X application-reliability analysis of data-rate and message-rate congestion control algorithms," *IEEE Commun. Lett.*, vol. 21, no. 6, pp. 1285–1288, Jun. 2017.

[36] C. Ayimba, V. Sciancalepore, C. Paolo, and M. Vincenzo, "I move U move: V2X-enabled wireless towing," in *Proc. CEUR Workshop*, 2022, pp. 1–9. [Online]. Available: https://dspace.networks.imdea.org/handle/20.500.12761/1634?show=full

[37] Y. Harkat and A. Amrouche, "Vehicle density, vehicle speed and packet inter-arrival time analysis in IEEE 802.11p EDCA based VANETs," in *Proc. Int. Conf. Signal Image Vis. Appl. (SIVA)*, 2018, pp. 1–6.

[38] W. Sun, H. Zhang, C. Pan, and J. Yang, "Analytical study of the IEEE 802.11p EDCA mechanism," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2013, pp. 1428–1433.

[39] P. Engelstad and O. Osterbo, "Delay and throughput analysis of IEEE 802.11e EDCA with starvation prediction," in *Proc. IEEE Conf. Local Comput. Netw. 30th Anniversary (LCN)*, 2005, pp. 647–655.

[40] M. Green, "'How long does it take to stop?' methodological analysis of driver perception-brake times," *Transp. Human Fact.*, vol. 2, no. 3, pp. 195–216, 2000. [Online]. Available: https://doi.org/10.1207/STHF0203_1

[41] H. Huang, H. Li, C. Shao, T. Sun, W. Fang, and S. Dang, "Data redundancy mitigation in V2X based collective perceptions," *IEEE Access*, vol. 8, pp. 13405–13418, 2020.

[42] S. Aoki, T. Higuchi, and O. Altintas, "Cooperative perception with deep reinforcement learning for connected vehicles," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, 2020, pp. 328–334.

[43] M. K. Abdel-Aziz, C. Perfecto, S. Samarakoon, M. Bennis, and W. Saad, "Vehicular cooperative perception through action branching and federated reinforcement learning," *IEEE Trans. Commun.*, vol. 70, no. 2, pp. 891–903, Feb. 2022.

[44] I. Ghnaya, T. Ahmed, M. Mosbah, and H. Aniss, "Maximizing information usefulness in vehicular CP networks using actor-critic reinforcement learning," in *Proc. 18th Int. Conf. Netw. Service Manag. (CNSM)*, 2022, pp. 296–302.

[45] U. Lang and R. Schreiner, "Managing security in intelligent transport systems," in *Proc. IEEE 18th Int. Conf. Intell. Transp. Syst.*, 2015, pp. 48–53.

[46] P. Škorput, H. Vojvodić, and S. Mandžuka, "Cyber security in cooperative intelligent transportation systems," in *Proc. Int. Symp. ELMAR*, 2017, pp. 35–38.

[47] J. Harvey and S. Kumar, "A survey of intelligent transportation systems security: Challenges and solutions," in *Proc. IEEE 6th Intl Conf. Big Data Secur. Cloud (BigDataSecurity), IEEE Int. Conf. High Perform. Smart Comput., (HPSC) IEEE Int. Conf. Intell. Data Secur. (IDS)*, 2020, pp. 263–268.

[48] M. R. Ansari, J.-P. Monteuuis, J. Petit, and C. Chen, "V2X Misbehavior and collective perception service: Considerations for standardization," in *Proc. IEEE Conf. Stand. Commun. Netw. (CSCN)*, 2021, pp. 1–6.

[49] C. Allig, T. Leinmüller, P. Mittal, and G. Wanielik, "Trustworthiness estimation of entities within collective perception," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, 2019, pp. 1–8.

[50] B. Hurl, R. Cohen, K. Czarnecki, and S. Waslander, "TruPercept: Trust modelling for autonomous vehicle cooperative perception from synthetic data," in *Proc. IEEE Intell. Veh. Symp. (IV)*, 2020, pp. 341–347.

[51] A. Shoker, P. Moertl, and R. Robles, "A first step towards holistic trustworthy platoons," in *Proc. IEEE 7th World Forum Internet Things (WF-IoT)*, 2021, pp. 795–800.

[52] P. Michalopoulos, J. Meijers, S. F. Singh, and A. Veneris, "A V2X reputation system with privacy considerations," in *Proc. IEEE 13th Int. Conf. Softw. Eng. Service Sci. (ICSESS)*, 2022, pp. 8–14.

[53] J. Godoy, V. Jiménez, A. Artuñedo, and J. Villagra, "A grid-based framework for collective perception in autonomous vehicles," *Sensors*, vol. 21, no. 3, p. 744, 2021. [Online]. Available: https://www.mdpi.com/1424--8220/21/3/744

[54] Y. Yin, T. Yu, K. Maruta, and K. Sakaguchi, "Distributed and scalable radio resource management for mmWave V2V relays towards safe automated driving," *Sensors*, vol. 22, no. 1, p. 93, 2022. [Online]. Available: https://www.mdpi.com/1424--8220/22/1/93

[55] R. Fukatsu and K. Sakaguchi, "Automated driving with cooperative perception using Millimeter-wave V2V communications for safe overtaking," *Sensors*, vol. 21, no. 8, p. 2659, 2021. [Online]. Available: https://www.mdpi.com/1424--8220/21/8/2659

[56] R. Fukatsu and K. Sakaguchi, "Automated driving with cooperative perception based on CVFH and Millimeter-wave V2I communications for safe and efficient passing through intersections," *Sensors*, vol. 21, no. 17, p. 5854, 2021. [Online]. Available: https://www.mdpi.com/1424--8220/21/17/5854

[57] A. Petrillo, A. Pescapé, and S. Santini, "A secure adaptive control for cooperative driving of autonomous connected vehicles in the presence of heterogeneous communication delays and Cyberattacks," *IEEE Trans. Cybern.*, vol. 51, no. 3, pp. 1134–1149, Mar. 2021.

[58] A. Coppola, D. G. Lui, A. Petrillo, and S. Santini, "Eco-driving control architecture for platoons of uncertain heterogeneous nonlinear connected autonomous electric vehicles," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 12, pp. 24220–24234, Dec. 2022.

[59] B. Caiazzo, D. G. Lui, A. Petrillo, and S. Santini, "Distributed double-layer control for coordination of Multiplatoons approaching road restriction in the presence of IoV communication delays," *IEEE Internet Things J.*, vol. 9, no. 6, pp. 4090–4109, Mar. 2022.

[60] G. Basile, D. G. Lui, A. Petrillo, and S. Santini, "ACC fuzzy-based control architecture for multi-body high-speed trains with active inter-cars couplers," in *Dependable Computing-EDCC 2022 Workshops*, S. Marrone et al., Eds. Cham, Switzerland: Springer, 2022, pp. 126–138. [Online]. Available: https://link.springer.com/chapter/10.1007/978-3-031-16245-9_10

[61] G. Fiengo, D. G. Lui, A. Petrillo, S. Santini, and M. Tufo, "Distributed robust PID control for leader tracking in uncertain connected ground vehicles with V2V communication delay," *IEEE/ASME Trans. Mechatronics*, vol. 24, no. 3, pp. 1153–1165, Jun. 2019.

[62] B. Caiazzo, D. G. Lui, A. Petrillo, and S. Santini, "Cooperative finite-time control for autonomous vehicles platoons with nonuniform V2V communication delays," *IFAC-PapersOnLine*, vol. 55, no. 36, pp. 145–150, 2022. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S2405896322030063

[63] A. Petrillo, A. Salvi, S. Santini, and A. S. Valente, "Adaptive multi-agents synchronization for collaborative driving of autonomous vehicles with multiple communication delays," *Transp. Res. C, Emerg. Technol.*, vol. 86, pp. 372–392, Jan. 2018. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S0968090X17303200

[64] L. M. Castiglione, P. Falcone, A. Petrillo, S. P. Romano, and S. Santini, "Cooperative intersection crossing over 5G," *IEEE/ACM Trans. Netw.*, vol. 29, no. 1, pp. 303–317, Feb. 2021.

CHRISTOPH PILZ received the M.Sc. degree in software engineering and management from the Graz University of Technology, Graz, Austria, in 2019, where he is currently pursuing the Ph.D. degree in affiliation.

He has been with Virtual Vehicle Research GmbH, Graz, Austria, since 2014, working across fields within the automated driving domain. Among others, creating a simulator for electronic control units in vehicles, testing perception sensor capabilities and performance, testing and verification of automated driving software, and analyzing, implementing, and verifying communication security.

PETER SAMMER received the B.Sc. degree in telematics from the Graz University of Technology in 2016, and the M.Sc. degree in information and computer engineering, focusing on embedded electronics from the Graz University of Technology in 2018.

He started working part-time with Virtual Vehicle Research GmbH in 2012 and full-time after his studies in 2018. He created multiple embedded data acquisition devices and sensor technologies. Since 2018, his focus has shifted to a higher level data acquisition and synchronization platform. In 2022, he became the Team Leader for data acquisition and visualization. His expertise is hardware design and embedded-C development.

ESA PIRI received the M.Sc. (Tech.) degree in electrical and information engineering and the Dr.Sc. (Tech.) degree in communications engineering from the University of Oulu, Finland, in 2008 and 2015, respectively.

He worked as a Researcher, most recently as a Senior Scientist with the VTT Technical Research Centre of Finland from 2007 to 2017. He joined a KNL Company to develop a novel radio access solution for the maritime industry from 2017 to 2019. Since 2017, he has been working as the Head of Product Management with Kaitotek Oy, Finland, developing new solutions for evaluating the quality and performance of communications networks. He has published over 30 journal articles, conference papers, and book chapters. His research interests and activities are mainly related to network performance measurement solutions, new network technologies, especially for mission-critical industry applications, communications in heterogeneous multi-access networks, and network management. He is participating in Technical Program Committees in Telecommunications and Network Conferences.

UDO GROSSSCHEDL received the B.Sc. degree in software engineering and management from the Graz University of Technology, Graz, Austria, in 2019. He is currently pursuing the M.Sc. degree in software engineering and management.

He worked with NXP Semiconductors, Gratkorn, Austria, from 2018 to 2019, in the Smart Card Toolchain Team, developing tools to program Smart Cards. In 2020, he worked with AVL List GmbH, Graz, Austria, developing tools and workflows for automated hardware and software integration tests. He has been with the Virtual Vehicle Research GmbH, Graz, since 2021, working in data acquisition and visualization, focusing on time-synchronized data acquisition systems and interactive data visualization of measurement data.

GERALD STEINBAUER-WAGNER received the M.Sc. degree in computer engineering and the Ph.D. degree in computer science from the Graz University of Technology, Graz, Austria, in 2001 and 2006, respectively.

He is currently an Associate Professor with the Institute of Software Technology, Graz University of Technology and leads the working group for Autonomous Intelligent Systems. His research interests include robot control, cognitive robotics, knowledge representation, reasoning, and model-based diagnosis. Moreover, he is the founding President of the Austrian RoboCup National Chapter and the Austrian IEEE Robotics and Automation Chapter. He is a member of the IEEE Robotics and Automation Society, ACM, and the Austrian Society for Artificial Intelligence.

LUKAS KUSCHNIG received the B.Sc. degree in electronics and computer engineering from the JOANNEUM University of Applied Sciences, Graz, Austria, in 2017, and the M.Sc. degree in information and computer engineering with a focus on embedded systems and visual computing from the Graz University of Technology in 2020.

He worked as a student assistant with NXP Semiconductors, Gratkorn, Austria, in 2016, within the Java Card Operating System Team, developing a tool to analyze Java Card EEPROMs automatically. From 2017 to 2020, he worked part-time with AVL List GmbH, Graz, Austria, as a Development Engineer in ADAS/AD, designing, and developing a mobile measurement system capturing dynamic ground truth. He has been with the Virtual Vehicle Research GmbH, Graz, Austria, since 2020, working across fields within the automated driving domain, focusing on sensor integration, environmental perception and simulation to extend and improve the automated driving platforms.

ALINA STEINBERGER received the B.Sc. degree in information and computer engineering from the Graz University of Technology, Graz, Austria, in 2021, where she is currently pursuing the masters's degree in information and computer engineering, focusing on robotics and embedded systems. She attended the technical upper-level secondary school focusing on electronic engineering and information technology in Klagenfurt, Austria.

She participated in several summer internships with Virtual Vehicle Research GmbH, Graz, Joanneum Research, Graz, and Infineon Technologies, Graz. She is currently affiliated with Virtual Vehicle Research GmbH for her master's thesis in the field of vehicle-to-everything communication.

MARKUS SCHRATTER received the M.Sc. degree in telematics from the Graz University of Technology, Graz, Austria.

He is currently a Researcher in the field of automated driving. He joined Virtual Vehicle Research GmbH in 2011, has been involved in several research projects with the automated vehicle demonstrators, and is the Team Leader for the embedded demonstrators. He joined Autonomous Racing Graz as a Technical Lead in 2019 and is responsible for the architecture of the software stack and integrating the different subsystems in the racing team. His research interests include automated driving, robotics, sensors, and integrating complex systems.