Large Data Transfer Optimization for Improved Robustness in Real-Time V2X-Communication

Alex Bendrick, *Graduate Student Member, IEEE*, Nora Sperling, *Graduate Student Member, IEEE*, and Rolf Ernst, *Life Fellow, IEEE*

***Abstract*—Vehicle-to-everything (V2X) roadmaps envision future applications that require the reliable exchange of large sensor data over a wireless network in real time. Applications include sensor fusion for cooperative perception or remote vehicle control that are subject to stringent real-time and safety constraints. Real-time requirements result from end-to-end latency constraints, while reliability refers to the quest for loss-free sensor data transfer to reach maximum application quality. In wireless networks, both requirements are in conflict, because of the need for error correction. Notably, the established video coding standards are not suitable for this task, as demonstrated in experiments. This article shows that middleware-based backward error correction (BEC) in combination with application controlled selective data transmission is far more effective for this purpose. The mechanisms proposed in this article use application and context knowledge to dynamically adapt the data object volume at high error rates at sustained application resilience. We evaluate popular camera datasets and perception pipelines from the automotive domain and apply two complementary strategies. The results and comparisons show that this approach has great benefits, far beyond the state of the art. It also shows that there is no single strategy that outperforms the other in all use cases.**

***Index Terms*—Data reduction, dynamic data profiles, large data objects, middleware, real time, wireless.**

I. Introduction

According to roadmaps [1], [2], [3], [4], future cooperative perception applications within the vehicle-to-everything (V2X) domain are envisioned with sharing of high-resolution camera or LIDAR sensor data or other large real-time (preprocessed) data, such as environmental maps augmented with sensor data. We refer to such large data objects as *samples S* that, due to their large size, have to be transmitted in a fragmented manner. Sensor data exchange is expected to enable and improve highly automated vehicles by providing additional insights into the vehicle environment, otherwise not obtainable by a single vehicle. Furthermore, such data exchange can also enable remote- or teleoperated driving services that either act as a stand-alone service [2] or as a backup for autonomous vehicles failing in challenging scenarios [5]. Meanwhile, the resolution and fidelity of the data produced by new sensors and sampling rates are increasing, thereby further raising demands on needed data rates for V2X communication.

Further complicating the issue are the safety constraints that apply to the transmission of each sample. In order to ensure *application resilience* as in the continuous safe operation of (highly automated driving) applications that rely on cooperative perception data, safety constraints must be respected: With sample transmission over a wireless channel being indispensable, the transmission reliability is a key concern, as wireless communication is inherently lossy. The sample transmission is also subject to real-time constraints as timing has a critical impact on the end-to-end latency of a perception pipeline, on the efficiency of sensor fusion and thereby on the ability to cooperate. Consequently, a sample must be completely transmitted prior to a sample deadline (D_S). Typically, application deadlines in perception pipelines are 100 ms, even for high sampling rates [6]. Thereby, *reliability* refers to the complete transmission of a fragmented sample prior to the deadline with no residual errors (missing fragments) remaining. To address the reliability constraints the wireless reliable real-time protocol (W2RP) [7] has been developed and shown to be highly effective in ensuring reliable sample exchange. However, transmitting large objects via a lossy wireless medium still poses significant challenges, as unrecoverable packet loss still can lead to violation of safety constraints if error rates exceed a certain threshold. Specifically, with channel conditions fluctuating, the transmission should offer sufficient *robustness* to maintain transmission reliability even under high error rates.

Purpose-built V2X wireless communication standards, such as the WiFi-based 802.11p/bd [8] and cellular V2X (C-V2X) [9] solutions, are primarily intended for the exchange of small messages, such as cooperative awareness messages (CAMs) or distributed environmental notification messages (DENMs) and therefore only offer fairly low data rates [10] that do not suffice for transmitting large samples reliably. Hence, to make the required data rates achievable using state-of-the-art technology we will assume 802.11ax as a representative high data rate standard. Due to the 802.11ax MAC layer protocol being very similar to 802.11p/bd [8], it serves as a proxy for a potential future 802.11-based V2X technology. Nonetheless, considering the number of vehicles and extrapolating potential data rate needs if all of those

Manuscript received 26 July 2024; accepted 28 July 2024. This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant ER168/32-2 and Grant ER168/35-1. This article was recommended by Associate Editor S. Dailey. *(Corresponding author: Alex Bendrick.)*

The authors are with the Institute of Computer and Network Engineering, Technische Universität Braunschweig, 38106 Braunschweig, Germany (e-mail: bendrick@ida.ing.tu-bs.de; sperling@ida.ing.tu-bs.de; ernst@ida.ing.tu-bs.de).

Digital Object Identifier 10.1109/TCAD.2024.3436548

vehicles take part in some form of V2X communication, *adaptation for improving robustness*, e.g., by reducing the required data rates and thereby allowing for more nodes to participate in V2X communication in a reliable manner will be beneficial.

So far, reliable V2X exchange of large samples relied on periodic streams of samples with a fixed size (cf. [7], [11]). Typically, publish–subscribe protocols, such as the data distribution service (DDS) [12], are used for that purpose. With DDS already being standardized for in-vehicle communication [13] it makes sense to extend DDS for use in wireless communication. However, various previous works have already shown that perception applications often rely only on a certain part of the data [14], [15]. Transmitting only those dynamically sized regions of interest (RoIs) can significantly decrease the volume of data that needs to be transmitted [14], [15], [16]. A second option to reduce data size that is widely used, e.g., in video encoding [17], [18], are forms of incremental updates. Considering an infrastructure camera that covers an intersection. At a given time, only parts of the image may change. Therefore, transmitting a complete sample every time results in the exchange of (potentially large amounts of) redundant data. By only transmitting incremental updates, the overall volume of data can be decreased. However, there is no single solution that is beneficial to all use cases and, to our knowledge, neither of the existing works on such mechanisms has focused on improving the reliability of wireless data exchange or evaluated its affects on wireless communication. To the contrary, some mechanisms, such as using video encoding, might actually hurt reliability in case of packet loss. Complicating the applicability of such mechanisms are dynamics V2X communication can be subject to. For example, in multicast scenarios, it might be necessary to transmit a complete sample once a new participant joins the applications. Hence, the protocol used for transmitting such data must be able to cope with such dynamics.

Contribution: In this work, we first extend W2RP to work with applications that transmit dynamically sized samples. Second, we integrate an application-agnostic data reduction mechanism right into W2RP that combines data optimizations with handling of errors and can adapt according to communication characteristics. We analyze the possibilities for robustness improvements using both mechanisms for V2X sample transmission. We analyze and compare real-world sensor data from a moving vehicle and a stationary infrastructure camera with respect to the applicability of data optimization mechanisms. The extended W2RP protocol has been implemented in OMNeT++ and evaluated using a remote driving use case and a scenario using an infrastructure camera that is supplying environmental information on an intersection area to nearby vehicles. We further investigate the performance using a physical demonstrator setup. The results show significant robustness improvements when using either of the two data optimization mechanisms, even in wireless channels affected by challenging burst errors and with dynamics requiring the occasional transmission of complete samples.

We start with review of related work in Section II followed by a short description of the channel and error model used within this work (Section III). The analysis of real-world camera data can be found in Section IV. Section V elaborates on the necessary modifications and extensions to W2RP followed by Section VI discussing the overhead introduced by either of the two options. The evaluation results from the simulation and the physical demonstrator setup can be found in Sections VII and VIII, respectively. We conclude this work in Section IX.

II. Related Work

A. Reliable Wireless Communication

Previous works have shown that the packet-level reliability mechanisms of state-of-the-art V2X standards, such as 802.11p/bd and C-V2X, do not suffice for protecting the transmission of large, fragmented samples, especially under consideration of real-time and safety constraints [7], [11]. With ultrareliable and low-latency communications (URLLCs) [19], [20] and wireless time-sensitive networking (TSN) [21], [22] there are two additional options for enabling reliable wireless communication. However, the focus of URLLC is on the reliable exchange of smaller (control) messages [20]. As a result, it offers a claimed reliability of 99.999% and a guaranteed latency of 1 ms for packets up to 32 B in size [19]. For still relatively small packets of up to 300 B the guaranteed latency increases to 10 ms [19]. With such guarantees, a maximum data rate of $300\,\mathrm{B}\cdot(1/10\,\mathrm{ms}) = 30\,\mathrm{kB/s}$ is feasible for reliable traffic. With single samples of high-resolution cameras or LIDARs ranging from multiple hundreds of kB to multiple MB in size, this fundamental limitation to small messages prevents URLLC from be applicable to such data exchange. Similarly, works on enabling TSN in existing wireless standards [23], [24] focus on the capabilities of exchanging small messages reliably. For this purpose, time synchronization and reliability mechanism are developed [21], however, these works lack considerations of the exchange of large time- and safety-critical samples entirely. Consequently, wireless TSN is not suited for cooperative perception applications.

The DDS- and real-time publish–subscribe (RTPS)-based middleware protocol W2RP [7] and its extensions have been developed to fill this void. The unicast W2RP protocol utilizes a backward error correction (BEC) protocol that focuses on optimal error protection for the exchange of fragmented samples under hard deadlines in a loaded wireless channel that is subject packet losses [7]. To further improve performance in scenarios with limited resources, an adaptive parameter selection approach for W2RP has been presented in [25] that proved effective in retaining reliability even under volatile channel conditions. The wireless multicast error protection protocol (WiMEP) [11] adds dedicated multicast functionality and the enhanced-W2RP (E-W2RP) [26] enables overlapping sample transmissions that proved highly effective in addressing burst errors. In the following, we do not differentiate between these protocol evolutions and refer to the combined functionality as W2RP. The protocols have been shown to be highly effective in enabling and improving reliable sample transmissions in simulation (cf. [7]) as well as on a physical demonstrator setup (cf. [11]). However, all of those three middleware protocols so far rely on static data profiles. As a result, neither of the protocols supports the reliable exchange of dynamically sized data inherent to the transmission of RoIs or incremental updates.

B. Data Optimization for Cooperative Perception Applications

In this article, we take video transmission for cooperative perception as an important use case for wireless transmission of large data objects. A popular way to decrease the data rate of video streams in consumer-focused use cases are video encoding standards, such as H.265 [17] or AV1 [18]. While mentioned as a potential option for the transmission of video streams in V2X applications [3] by the 3rd generation partnership project (3GPP) (standardization body responsible for cellular broadband communication), there are significant issues that impede usage of such data in safety-critical cooperative perception applications. Such video encoding codecs are lossy in particular in scenery changes with large video frame differences, which is particularly relevant for perception. While they can adapt to worsening channel conditions by decreasing fidelity, which is a potential problem for safety of perception applications in itself, there are no means of counteracting the inevitable packet loss that will occur when transmitting data over a wireless channel. Packet loss in turn can result in consecutive samples being affected by errors such as image artifacts that would significantly harm the safety of the intended function (SOTIF) and diminish application resilience. Furthermore, cooperative perception applications require all frame types to adhere to timing constraints, even the occasional I-frames containing complete image data. Given the large size, deadline violations can be expected, negating any potential benefit from using video codecs. As a result, video codecs are not suitable for use in cooperative perception applications that require the exchange of sensor data via a lossy wireless channel.

RoI-based data optimization mechanisms originate in sensor data processing (processing on smaller data is faster) and could solve some of these problems, by removing unnecessary data but keeping and transmitting the region of interest in a lossless manner. In [14] and [15], using an in-vehicle traffic light detection as an example, it was shown that the size of the transmitted information could be significantly reduced. Using data from the Autoware simulator AWSIM, Sperling et al. [15] showed a reduction in sample size by a factor of more than 76 when exchanging only RoIs (maximum size 36.3 kB) compared to exchanging the whole camera frame (2.76 MB). Sperling et al. [15] combined the RoI-based data optimization with a middleware architecture that replaces the common unidirectional data-centric communication design of DDS/RTPS-based communication toward a (self-)adaptive and subscribe-centric design, in which subscribers can request (specific parts of) data in a caching-like approach, however, this does not limit applicability of such mechanisms to exclusively publish–subscribe-based protocols such as W2RP. While mentioning wireless communication as a use case for such an RoI-based data optimization mechanism [15], its effectiveness when transmitting RoIs over lossy wireless networks and its effects on robustness in such scenarios have not yet been discussed.

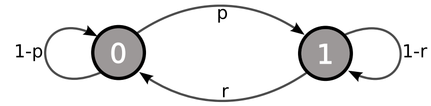


Fig. 1. GE model used to model burst errors.

A DDS implementation that integrates its own dedicated data reduction mechanism can be found in real-time innovation's (RTI) Connext DDS. Specifically, Connext DDS introduces TypeCode, a scheme that allows to partition data into multiple fields, with only a selection of fields being transmitted, thereby reducing the overall volume of data that is exchanged between nodes [27]. There are notable disadvantages of using TypeCode fields, with 1) specific user and developer interaction being required to create specific topics and data structures that support and use TypeCodes and 2) being limited to such specifically partitioned data structures. Transmitting RoIs from an image is not feasible using TypeCode fields. While certainly an interesting option to reduce the amount of data transmitted, TypeCode fields lack the flexibility of being applicable to optimize the data size of arbitrary samples in cooperative perception applications.

III. Channel and Error Model

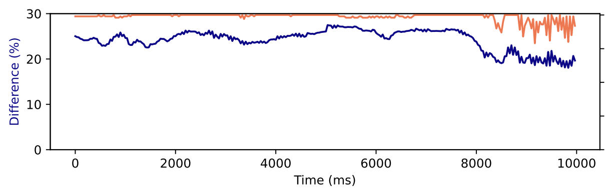
Fading effects, such as reflections, shadowing, and potential collisions, result in wireless communication being inherently lossy. In nonoverloaded channels, typically, the first source for errors—bit-error-related fading effects—dominates. In this work, we follow [7] and [11] in using a bit error rate (BER) to model those errors. In 802.11 and cellular wireless communication, there are means of reducing the impact of bit errors by means of forward error correction (FEC), however, in the following, we refer to the residual error rate as experienced after FEC. Using the packet size S_p, a frame error rate (FER) can be derived from the BER using the following equation:

$$\mathrm{FER}(S_p, \mathrm{BER}) = 1 - (1 - \mathrm{BER})^{(S_p \cdot 8)}. \quad (1)$$

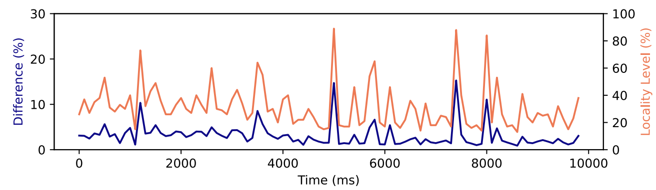
We further use a two-state Gilbert–Elliot (GE) model [28], [29] to describe errors affecting consecutive fragment transmissions. The GE model comprises a good state 0 and a bad state 1 (cf. Fig. 1). The transitions between the two states are modeled using the probabilities p (error occurs following a successful transmission) and r (and error is followed by a success). Using appropriate parameter selections, the GE model can be used to model burst errors which is not possible using Bernoulli experiments to model individual bit or frame errors such as done in [7].

IV. Feasibility of Data Transfer Optimization for Cooperative Perception Applications

In this section, we present an analysis of real-world camera data from 1) a moving vehicle and 2) a stationary infrastructure camera covering the intersection area of a public

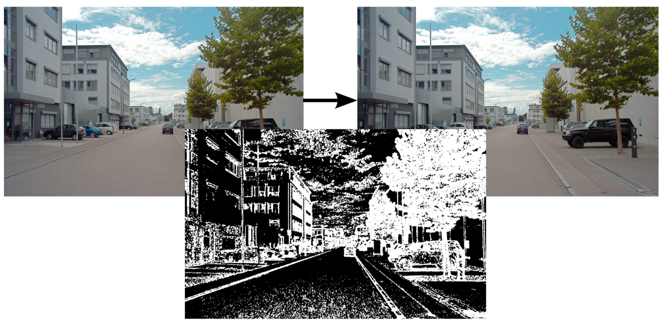


(a)



(b)

Fig. 2. Comparison of differences between consecutive camera frames of a (a) moving vehicle and (b) stationary infrastructure covering an intersection area. Additionally, the locality level that quantifies how many parts of the image are affected by changes/updates is displayed. There, 100% corresponds to updates being distributed across the whole image whereas smaller values correspond to updates being limited to less spread area of the image. (a) Analysis of differences and their locality levels in camera data from a moving vehicle, courtesy of the A2D2 dataset [30]. (b) Analysis of differences and their locality levels in camera data from a stationary infrastructure camera, courtesy of data from [31].



(a)

(b)

Fig. 3. Excerpts from the A2D2 [30] and DVS [31] datasets. A sequences of two frames per dataset and a mask visualizing all area in which updates/changes occurred between the two frames are displayed. The update mask from the camera data of the moving vehicle shows updates across the whole image (high locality level) whereas updates in the infrastructure camera are limited to the actual cars driving on the road, with the rest of the image remaining unchanged. (a) Sequence of two frames and mask highlighting all differences between the two frames from the A2D2 dataset. (b) Sequence of two frames and mask highlighting all differences between the two frames from the DVS dataset.

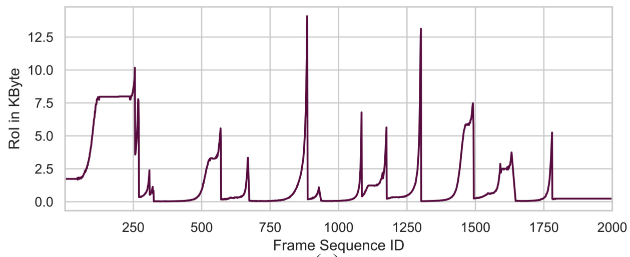
road to determine the feasibility of different data optimization techniques. For 1), we used data from the Audi Autonomous Driving Dataset (A2D2) [30], captured during a ride through Ingolstadt, Germany. For the infrastructure camera data 2), we used camera data from [31] that covers a intersection in a small city in the U.S. We further demonstrate the scope of the size and dynamism of contextual RoI-based object communications by showing simulated data from AWSIM[1] and the Autoware Software Stack [32] (cf. Fig. 4).

To determine what kind of data optimization technique is feasible in the given scenarios we start with determining the differences between two consecutive frames in both the mobile and stationary scenario. Additionally, we compare the locality level of the previously addressed differences to answer the question of “How far are differences distributed across the image?”. For this purpose, camera images are distributed into 100 equally sized parts. The percentage of parts that are subject to differences between two consecutive frames is defined as the locality level. Consequently, smaller locality levels correspond to differences being limited to a smaller portion of the image whereas 100% corresponds to differences being distributed across the whole image. Exemplary excerpts from the datasets that show two frames each and the differences between those are shown in Fig. 3.

[1]https://github.com/tier4/AWSIM

Analyzing the A2D2 dataset, it becomes apparent that, due to the vehicle moving through the environment, consecutive frames show a moderate percentage of differences (roughly 20%–25% on average) as can be seen in Fig. 2(a). However, the locality level always remains at or close to 100%. This also becomes apparent in the visual representation of the difference in Fig. 3, with differences being distributed across the whole image. Consequently, any data optimization mechanism that tries to reduce the amount of data by only transmitting incremental updates would be inefficient for scenarios in which the sensor is moving itself. Instead, other mechanisms, such as transmitting RoIs, might result in better performance.

RoIs can be predetermined by the processing pipeline by, e.g., tracking objects and traffic signals. In the Autoware processing pipeline, the traffic light recognition module does not process camera frames as a whole, but predetermined areas that were calculated via internal tracking structures. Fig. 4 shows the data volume of these RoIs along a 200-s drive in the AWSIM simulator. The simulated camera publishes camera frames with the resolution 1280×720 [see Fig. 4(b)], resulting in raw frame sizes of 2.76 MB. It can be clearly seen that traffic light RoIs are highly dynamic, but their upper size is bounded by the traffic light’s size and its minimum distance to the vehicle. As a result, RoI sizes lie within [0.5 kB, 13 kB] which, on the upper end, corresponds to less than 0.5% of the original data. Furthermore, their data size is predictable, as the size of a RoI increases with the vehicle getting closer to the



(a)


(b)

Fig. 4. Example of camera data from a moving vehicle (here from AWSIM). RoIs typically only make up a small fraction of the whole image. (a) RoIs sizes for a sequence of camera frames. It is apparent that sizes are highly dynamic, with the size increasing the closer the vehicle is to the traffic light. (b) Screencapture of simulated camera data of AWSIM with traffic light RoIs as recognized by the perception algorithms of the Autoware software stack. The different sizes of the two RoIs are apparent, with the one further ahead being less than a quarter the size of the one right in front of the vehicle.

respective object, before they vanish from the camera's field of view. RoIs have to be tracked individually as multiple RoIs can appear per frame and their size and position within the camera frame is ultimately strongly dependent on the speed and overall movement of the vehicle. While incremental update mechanisms are not suitable for use cases streaming sensor data out of moving vehicles, RoIs could prove highly effective in improving such data exchange. Furthermore, with RoIs also being used in LIDAR data processing [33], advantages would not only be limited to camera data.

In contrast to the camera data of the moving vehicle, the comparison of consecutive frames from the stationary infrastructure camera of [31] shows that on average only 5%–10% change on a per frame basis [cf. Fig. 2(b)]. Furthermore, the changes are typically limited to 20%–50% of the whole image. This is the result of only those parts of the image changing which contain vehicles moving across the road. This is also visualized by the differences between the two camera frames in Fig. 3(b). As a result, an incremental update mechanism could significantly reduce the total amount of data that needs to be transmitted in such a scenario—or in other use cases that make use of stationary infrastructure cameras, e.g., in automated valet parking. However, as can be seen in Fig. 2(b), there is still a potential for rare outliers which can be caused by sudden changes within the environment. Such instances still require the transmission of large portions or even the complete sample occasionally. Thus, any incremental update mechanism must be able to cope with such occasional dynamics.

Concluding this section, the analysis of the video data from the mobile and stationary scenario showed that there is no single optimization strategy that addresses all use cases. Instead, incremental update mechanisms and RoIs techniques can be applied to different use cases. Consequently, with there being no one-size-fits-all solution, we will adapt W2RP to handle both the transmission of RoIs and incremental updates.

V. Data Transfer Optimization Strategies

In both use cases of cooperative perception, the data streams are periodic with a fixed sample deadline. In the following, we also use network standards where the channel data rate is sufficient for a transmission of all samples within their deadline, as long as there are no frame losses. Rather than trying to increase the data rate, the dynamic data size minimization presented in this article is used to maximize slack for W2RP frame error correction, and, thereby ideally improve robustness. As we will see, W2RP is capable of accommodating the extra slack for reliable sample transmission up to high FERs that are unmanageable for existing transmission protocols, including static data configurations of W2RP.

Based on the previous analysis of different data, it becomes apparent that there is no single solution for data optimization. With the goal of W2RP being widely applicable to different use cases, we will present two different solutions. However, neither the RTPS-standard nor W2RP support the transmission of samples with dynamic size. Consequently, we extend W2RP to allow for such dynamic data profiles. The extensions are twofold. First, we add an RTPS-level incremental update mechanism that is data-agnostic, with overlapping sample transmissions being used to handle dynamics in which complete samples are still needed occasionally. Thereby, infrastructure camera data transfer, such as [31], can be minimized. Second, the adaptations used for the incremental update mechanism are exploited to enable W2RP to cope with applications transmitting dynamically sized samples, e.g., RoIs.

A. RTPS-Level Incremental Sample Updates

The main objective of the incremental update mechanism is to only transmit the information that are not already known at the reader(s). Thereby, the effective amount of data exchanged via the wireless channel is reduced. The channel resources released by decreasing effective sample sizes can then be used for improved robustness to higher error rates by using the additional slack for more retransmissions, or increasing the sampling rate or resolution of samples.

With DDS/RTPS using a history buffer that holds multiple revisions of a sample, there is information available on previous samples. Once a new sample is added to the history, it is forwarded to the writer that fragments the sample and afterwards transmits it to all subscribed readers. Here, during the fragmentation process, by comparing each fragment of the new sample with the respective fragment of the previous sample, *updated fragments* are determined and are marked

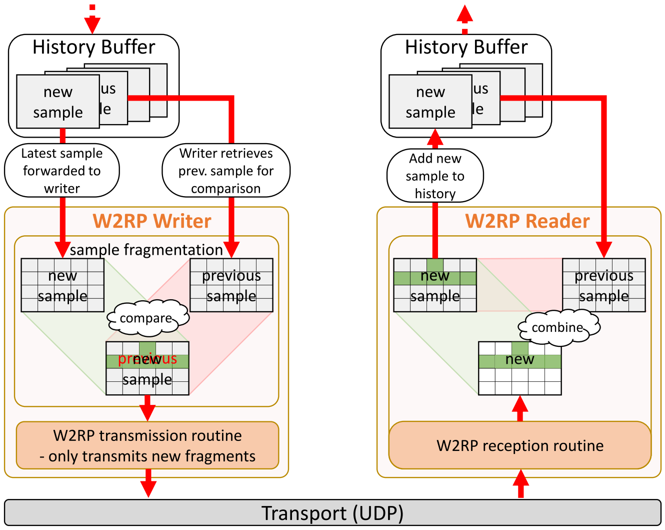


Fig. 5. Incremental update mechanism added to W2RP. The previous samples found in the history buffer are used to determine what fragments have changed (writer). At the reader, fragments marked as new are combined with the residual fragments from the previous sample.

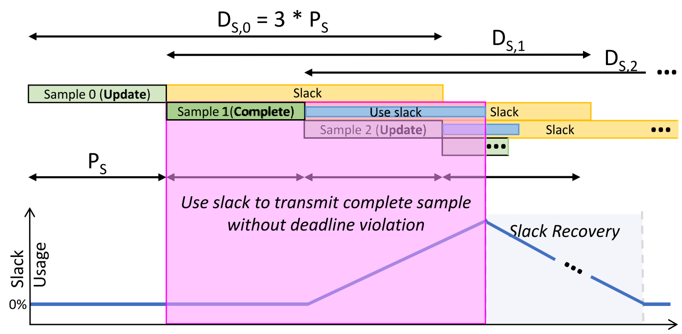


Fig. 6. Using slack to accommodate the transmission of complete samples. After the successful transmission of a complete sample, the slack usage recovers to regain robustness to errors or future needs to transmit complete samples.

accordingly as visualized in Fig. 5. When transmitting the sample, first, the decision whether the complete sample or only the updated fragments shall be exchanged is made. The former might be necessary in multicast scenarios if a new reader joined the application which consequently requires the complete sample once, prior to incremental updates being feasible.

On the receiving side, the reader must be able determine which fragments it is expected to receive in case of incremental updates being exchanged. For this purpose, we define a new RTPS submessage called UPDATE that, if appended to the fragment transmission, signals to the reader that only incremental updates will be transmitted. The UPDATE submessage contains a bitmap that marks all *updated fragments* the reader shall expect during the current sample's transmission. Using this information, the reader combines its previous sample with the *updated fragments*, resulting in the reception of the complete sample. In case of errors resulting in packet loss, the reader also is aware of what fragments it is still missing, hence can provide accurate *NackFrag* feedback to the writer, ensuring the correct retransmissions can be performed.

The mechanism must be able to cope with the occasional need to transmit complete samples without any deadline violations (cf. Fig. 6). The overlapping sample transmission previously exploited for robustness against burst errors [26] can be used for this purpose. If a complete sample shall be transmitted, the writer exploits its slack that overlaps the subsequent sample transmissions. With subsequent samples only requiring incremental updates the slack usage recovers, again leaving enough slack for transmitting a complete sample or performing additional retransmissions in case of burst errors. Thereby, loss-free transmission can be guaranteed if complete sample transmission and/or burst errors do not occur to close to each other.

Compared to higher-level mechanisms that use incremental updates, e.g., video encoding codecs, the integration in W2RP has significant advantages with respect to reliability. There is no pre- or post-processing that extends the critical transmission latency, and all available slack is used for error correction. The smaller the update, the higher the robustness. However, while being easy to use, there are certain limitations to the presented approach. First, the binary representations of samples have to be aligned in the same way the data structures exchanged are designed. Second, the mechanism will be ineffective for samples such as camera frames from a moving vehicle (cf. Section IV). However, e.g., transmitting a camera stream of a static infrastructure camera that covers an intersection area and thereby, only parts of the frame changes with each sample serves as a notable example where the incremental update mechanism could be highly effective. Third, the granularity of the incremental updates is dictated by the fragment size.

B. RoI-Based Sample Exchange

Unfortunately, RTPS, and thereby also W2RP, cannot determine RoIs. While the application-awareness allows W2RP to have certain knowledge of kind of samples are transmitted, it works solely on the fragment representation. Each fragment contains a serialized representation of the respective data. Thus, operations that require contextual knowledge of the data, such as determining RoIs, are not feasible. Nevertheless, W2RP is still a viable option to transmit RoIs reliably if the RoI determination is done on a different layer.

Some minor modifications are needed to achieve this. First, the middleware has to allow dynamically sized samples and forward those unchanged to W2RP. While not considered in the DDS [12] and RTPS [34] specifications, this can be assumed to be a trivial modification.

The critical issue lies in notifying readers on how many fragments to expect for a given sample. For this purpose, the UPDATE submessage introduced above can be repurposed. When enabling the transmission of dynamically sized samples, the bitmap within the UPDATE submessage highlights all fragments the current sample is comprised of. Consequently, the reader can determine that it has received the complete sample regardless of the current sample size that might change for each sample. In contrast to the incremental update mechanism however, the new data shall not be combined with previous data. Therefore, an additional RoI *configuration* bit

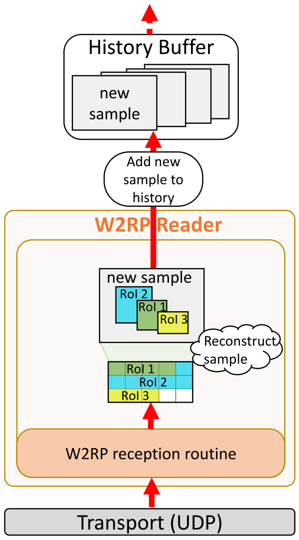


Fig. 7. If the RoI *configuration* bit is set, the reader determines which data is relevant based on the UPDATE submessage. Based on those information, RoIs are reconstructed, resulting in a list of RoIs that is passed to the applications via the history buffer.

is added to the UPDATE submessage. If the bit is set to *1*, the RoIs are directly reconstructed from the transmitted serialized data and are added to the history buffer (cf. Fig. 7). Thereby, applications are granted access to the (list of) RoIs.

VI. Overhead of Proposed Mechanisms

As both complementary options presented in Section V require processing that was not needed before, additional overhead is introduced that reduces the slack and, hence, could compromise reliability and robustness if delays would be too long. In the following, we analyze computation overhead of both the incremental update mechanism as well as the existing RoI determination mechanisms in the Autoware software stack, as described above. The data has been measured on the same physical demonstrator setup that will be used for physical proof-of-concept experiments in Section VIII, with hosts comprising an Intel Core i5-11500 and 16 GB of RAM. As an operating system, Ubuntu 20.04 was used.

A. Incremental Updates for RTPS

The incremental update mechanism is integrated at the RTPS-level fragmentation process. We chose the open-source FastDDS[2] implementation as a basis for our evaluation. We start with a quick analysis of the existing fragmentation mechanisms, then modify it in order to allow to determine which fragments did change and require an update to be sent.

The sample fragmentation process in FastDDS is performed on already serialized data. Based on the configured fragment size, the sample is then simply divided into smaller pieces. The proposed incremental update mechanism is integrated with the existing fragmentation process. Each fragment is basically just a serialized representation (byte array) of a certain part of the

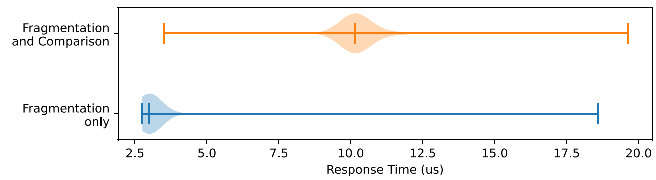


Fig. 8. Benchmark of the fragmentation mechanism in FastDDS against a modified one that also determines difference to the previous sample. The time needed to fragment (and compare) a complete sample is visualized. The sample and fragment size have been set to 200 and 10 kB, respectively.

data. Using *memcmp*[3] such data can be compared efficiently. We benchmarked the existing fragmentation process against the modified one, using 200 kB large samples that are divided into 10 kB fragments. A total of 10 000 randomly generated samples have been processed for each option.

The results are visualized in Fig. 8. On average, the existing fragmentation process in FastDDS takes approximately 3.75μs to complete, however, rare outliers that take up to 20μs exist. In contrast, the additional comparisons result in the average response time of the process being tripled (10μs). The outliers, however, only marginally exceed those measured previously for the unmodified fragmentation (only) process. Despite the average response time increasing by 200%, only making the transmission of a single fragment unnecessary already leads to the incremental update mechanism being beneficial. Neglecting all protocol overheads that affect all fragment transmissions, transmitting a single 10-kB fragment using a 600-Mb/s channel takes 133μs (pure transmission time without channel arbitration). Comparing this to the increase response time for the additional comparison needed for the incremental update mechanism, it becomes apparent that the introduced overhead is negligibly small.

Furthermore, adding a 8-B UPDATE submessage capable of addressing a total of 256 fragments per sample also only introduces a marginal overhead of less than 1%, considering the fragment size typically exceeds 1000 B. Consequently, the potential benefits of the incremental update mechanism outweigh the small overhead.

B. RoI Determination

RoIs are already determined by autonomous driving stacks such as Autoware (cf. [32]) for use in in-vehicle functionality. Consequently, it is not complicated to integrate V2X functionality that transmits RoIs wirelessly to other vehicles, the infrastructure (edge) or the cloud.

In Autoware, RoIs are not determined on each camera frame in a sequential manner, but rather concurrently to the sampling of sensor data. Through tracking RoIs over time, an estimation of the size and position of RoIs is possible independent of the actual camera frame. While detailed description of the RoIs determination mechanisms are out of scope here, the timing is still relevant.

Given the concurrent nature of the RoIs determination, there are two aspects to consider. First, there is the need

[2]https://github.com/eProsima/Fast-DDS

[3]https://en.cppreference.com/w/cpp/string/byte/memcmp

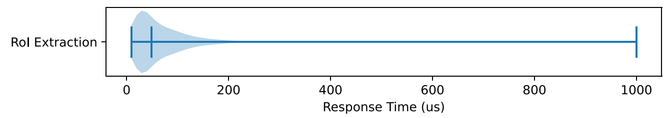


Fig. 9. Benchmark of RoI extraction process in the Autoware software stack. The time needed to extract multiple, up to 13 kB large RoIs from 2.76 MB samples is visualized.

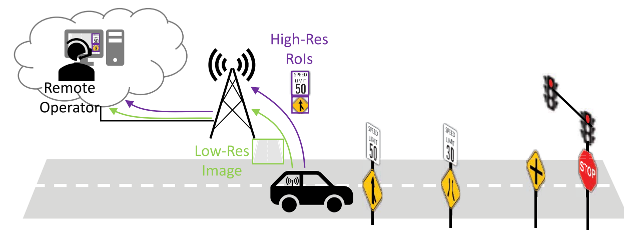


Fig. 10. Autonomous vehicle supervised and controlled by a remote operator. Given uncertainties of the detection algorithms of the vehicle, e.g., in bad weather conditions, basic information on traffic signs and lights RoIs are continuously transmitted to a remote operator. There are two options: either transmit the complete samples (2.76 MB) or only transmitting a low-resolution image (approximately 60 kB) in combination with high-resolution RoIs (2–14 kB).

for synchronization of RoIs and camera frames and, second, the actual RoI data needs to be extracted from the complete sample. As RoI determination can be adapted to match the timing and rate of the camera stream, we measured virtually zero timing overhead with respect to the synchronization. The only notable overhead introduced by RoI mechanisms is related to the extraction of the RoIs from the original image. We measured the time it takes for this operation to execute during a run of AWSIM. The RoI extraction is computationally more expensive than the determination of incremental updates. On average, Fig. 9 shows the RoI extraction taking approximately 50μs to complete, frequently also reaching up to 200μs. In general, longer response times do not correlate with a larger number of RoIs that are extracted from a given image. Rare outliers can even take up to 1 ms, however, based on data and source code analysis, these outliers do not seem to be data dependent but rather caused by operating system or middleware interference. Despite these uncertainties, we still consider these outliers as the worst case. Nevertheless, the massive reduction in size of the transmitted sample due to only transmitting RoIs still far outweighs even the worst case overhead. Using the example from above where a 10-kB fragment transmission takes 133μs in a 600-Mb/s channel, it only takes a reduction of eight fragments, i.e., less than 3% of the total sample size to compensate for the software timing overhead. The single 8-B UPDATE message at the beginning of a fragment transmission also has a negligible effect.

VII. Evaluation

The following section comprises the simulative evaluation of the presented protocol. For this purpose, we utilize a OMNeT++ simulator courtesy of [11]. The simulator has been configured to use 802.11ax as a proxy for a future V2X technology for high data rates as state-of-the-art V2X standards are incapable of the data rates needed for the exchange of cooperative perception data. We investigate two scenarios: First, camera data is transmitted from an autonomous vehicle to a remote operator. In this use case, the human operator uses the transmitted camera stream to supervise and control vehicle operation. Resilient application service (supervision of the moving vehicle) is highly critical and requires reliable high-resolution sensor data even under challenging conditions. The two major challenges are noisy V2X channels and difficult scenes, such as bad weather conditions. In bad weather conditions, uncertainties with respect to the vehicle's own perception mechanisms exist, such as incorrect traffic sign detection or a wrong object classification, where human perception monitoring could be requested. One way to master remote control even under coincidence of both challenges could be high-resolution camera transfer of RoIs covering potentially interesting objects, as in [32], combined with lower resolution of the total scene. The second scenario is based on a infrastructure camera that covers an intersection area within a parking garage used for automated valet parking. The duration of the simulated scenarios was 3600 s per experiment.

We analyze both use cases with respect to reliability and robustness, comparing the effects of transmitting RoIs or incremental updates, respectively. To reiterate, within the scope of this work, we refer to reliability as the protocol's capability to facilitate successful sample exchange within the sample deadline D_S and not as reliability as a statistical parameter. Robustness then describes a protocol's capability to adapt to higher error rates.

A. Data Exchange From Within Moving Vehicle

There are multiple options for data exchange originating from within a moving vehicle, e.g., for use in a remote operation use case such as visualized in Fig. 10. Using existing state-of-the-art technologies, complete samples would need to be transmitted using MAC layer retransmissions. With recent middleware protocols such as W2RP, it can be expected to improve upon MAC layer retransmissions. Finally, we investigate the performance of RoIs and the incremental sample update mechanisms. For this purpose, we test in a channel with 600 Mb/s.

Camera data is transmitted from the vehicle with a sampling rate and sample deadline of 100 ms. Using AWSIM data as a reference, a complete sample accounts for 2.76 MB. The parametrization of the RoI and incremental update mechanisms are based on the analysis of datasets in Section IV.

If low error rates permit, complete samples are transmitted. Under higher FERs, one of the two complementary data rate minimization methods is applied. For the RoI-based approach, a low-resolution (320×180 pixels), grayscale version of the full scene (60 kB) is always streamed to the remote operator. While enough to control basic driving maneuvers, to adhere to given speed limits and right-of-way rules (and classify potential conflicting objects), high-resolution data on traffic signs and lights are needed as those information can get lost in a low-resolution image or misinterpreted by the vehicle itself. For this purpose, the continuous exchange of high-resolution RoIs is also required. As stated in Section IV, RoI

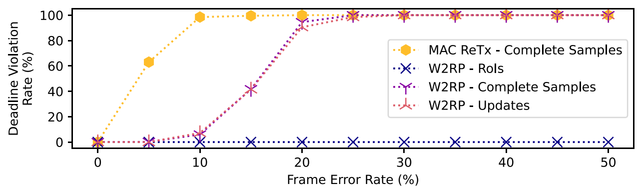


Fig. 11. Comparison of different data optimization schemes (no optimization, incremental updates, and RoIs) in a scenario where camera samples are exchanged between moving vehicles via a 600-Mb/s channel. The robustness of sample transmissions is evaluated using increasing FERs.

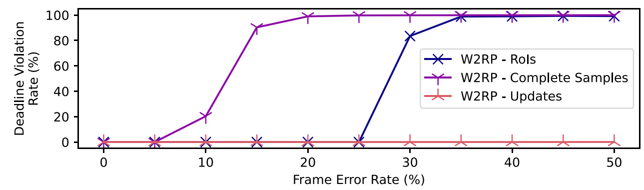


Fig. 12. Robustness of the exchange of infrastructure camera data is evaluated using different optimization approaches in a channel with a data rate of 400 Mb/s and increasing error rates.

sizes lie within [0.5 kB, 13 kB], with the size increasing when approaching a traffic sign/light. With the addition of traffic signs multiple RoIs (here up to 10) can be found per sample.

Incremental updates can only marginally reduce the amount of data that needs to be transmitted, because of the rapidly changing camera contents of the moving vehicle. Based on the locality level in Fig. 2(a), we chose an update ratio of 95% and 98% of samples that still need to be transmitted completely.

The results are visualized in Fig. 11. At 600 Mb/s, the data rate is nominally sufficient to transfer all samples completely. However, the limited number of retransmissions, as specified for the application-agnostic MAC layer protocol, cannot completely avoid packet losses even at low FER, such that the transmission remains incomplete and the sample misses its deadline. With application level BEC, as in W2RP, the slack from initial transmission to deadline can be completely used for BEC, leading to moderate robustness. This gain in robustness confirms the results from previous works [7], that packet-based MAC layer retransmissions are ineffective at ensuring timely and reliable exchange of large samples. With application level BEC, as in W2RP, the slack from initial transmission to deadline can be completely used for BEC. The results are much better, but the robustness is still moderate, because of the limited slack for error correction within the deadline at substantial channel load (ca. 50%). Again, as expected, there is no significant difference between complete sample and incremental update transmission using W2RP. Consequently, the experimental results verified that an incremental update mechanisms is not applicable to a scenario in which a camera from a moving vehicle shall be transmitted. In contrast, the RoI approach proved highly effective in ensuring reliable sample transmissions and thereby significantly outperformed default W2RP and the incremental update mechanism with respect to robustness.

B. Data Exchange Using Static Infrastructure Camera

In the following section, we investigate the performance of the three mechanisms for the exchange of a camera stream from a static infrastructure camera. Here, we use the data from [31] as a point of reference for a infrastructure camera. Therefore, a complete sample is roughly 675 kB in size. Camera images are sampled with a frequency of 30 Hz to improve object tracking performance. Meanwhile, the sample deadline remains unchanged at 100 ms [6]. E-W2RP is used to allow for overlapping sample transmissions and benefit from the extended deadline compared to the sample period. Details on E-W2RP can be found in [26]. A 400-Mb/s channel, that is nominally sufficient for transmitting such data in error-free scenarios, is used for this purpose. Again, we test transmitting the complete sample, incremental updates and RoIs at increasing FER. For the incremental update mechanism to accurately model the data dynamics of the infrastructure dataset [cf. Fig. 3(b)], 5% of samples need to be transmitted completely. This results in rare data rate spikes as it would happen for samples with high locality levels where large potions of the sample need to be transmitted as part of the incremental update.

Determining RoIs is not as straight-forward as for a traffic light and sign detection as used as in the previous experiment. The potentially interesting part of a camera frame comprises all vehicles and other traffic participants, regardless of whether moving or stationary. This way, a vehicle receiving that data has complete knowledge about all potential sources for hazards at a given intersection. Consequently, we define RoIs as bounding boxes around all vehicles, cyclist and pedestrians. Again, depending on the distance to the camera, the size of the RoIs varies, with the maximum size of RoI being approximately 20 kB. Furthermore, multiple (here between 15 and 25) RoIs can be present in a single frame. All of those RoIs have to be transmitted. The simulated incremental update mechanism has been configured based on the data presented in Fig. 2(b). Using the locality level as a reference, on average only 30% of the sample need to be exchanged as part of an incremental update. However, dynamics that result in locality level spikes requiring the transmission of larger portions of a sample or even a complete sample occasionally have also been considered.

The results in Fig. 12 clearly show the incremental update mechanism performing the best with respect to robustness as none of the tested FER lead to deadline violations. Notably, this remains true despite the need to occasionally transmit samples completely. While outperforming the transmission of complete samples with respect to robustness, RoIs are not the optimal solution for such an infrastructure camera use case. As stationary vehicles must also be transmitted, the data reduction is not as effective as the incremental update mechanism. Whereas RoIs can be exchanged reliably up to FER of 25%, transmitting complete samples using W2RP only works fine for FER smaller than 5%. Again, the simulation results showed that depending on the use case, there is no single data optimization solution but rather different approaches must be deployed for optimal robustness. Here,

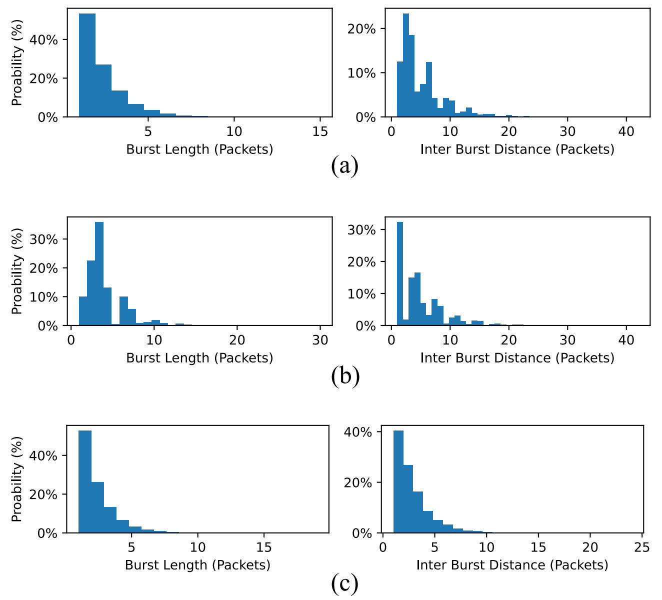


Fig. 13. Burst error characteristics for all three GE model configurations. Both the burst length and interburst distance are displayed. (a) *Baseline:* GE parameters $p = 0.18$ and $r = 0.5$. (b) *Longer bursts:* GE parameters $p = 0.18$ and $r = 0.3$. (c) *More bursts:* GE parameters $p = 0.3$ and $r = 0.5$.

as expected, the incremental update mechanism managed to improve robustness the most, clearly outperforming RoI-based transmissions.

C. Data Transfer Optimization in Burst Error Scenarios

So far, only uniformly distributed errors have been considered. However, often wireless channels are subject to burst errors [35], that increase the burden on the protocol to ensure reliability. Hence, to finalize the simulative evaluation, we use a burst error scenario as a stress test of the proposed mechanisms under extremely difficult conditions.

A GE model has been used to model burst errors in simulation. As a *baseline*, the parameters ($p = 0.18$ and $r = 0.5$) have been adopted from [26], resulting in an average error rate of 29% during the experiments. We then also test with *longer bursts* ($p = 0.18$ and $r = 0.3$) and *more bursts* ($p = 0.3$ and $r = 0.5$), with both configurations resulting in an average error rate of 45%. Fig. 13 visualizes the burst error characteristics of the three configurations, as captured from the experiments, with respect to the burst length and interburst distance. Thereby, the *longer bursts* configuration in Fig. 13(b) clearly shows the increased burst length compared to Fig. 13(a) with a maximum burst of 40 consecutive errors being observed and the whole distribution being shifted toward longer burst lengths. Similarly, Fig. 13(c) visualizes a decrease in inter burst distance compared to the other two GE configurations. Otherwise, the setup is identical to the infrastructure camera setup in Section VII-B.

Instead of only evaluating the robustness in burst error scenarios we also investigate to what degree the proposed data transfer optimization mechanism might potentially improve application performance by reducing latencies. This would allow for faster object detection and subsequently faster

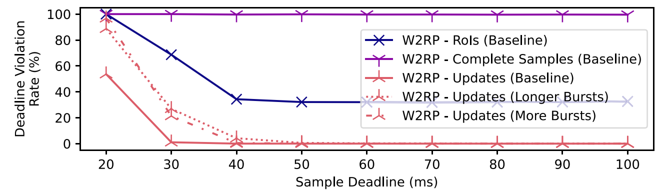


Fig. 14. Reliability of different sample transmission approaches in the infrastructure camera scenario. Here, the wireless channel is subject to bursts errors. Different sample deadlines have been tested.

reaction time, thereby improving application quality of service (QoS). To test this hypothesis, we reduce the sample deadline in 10 ms increments starting at 100 ms and evaluate whether reliable sample transmission is still feasible. If smaller sample deadlines still allow for reliable sample transmission, latencies are lower, thereby improving application QoS.

Given that the transmission of RoIs and complete samples did not work for uniformly distributed errors at error rates of 29%, it is expected that neither will be able to ensure reliable sample transmission in the evaluated (baseline) burst error scenario. Fig. 14 clearly confirms this assumptions. As a result, no further experiments have been performed with either mechanism and even more challenging burst error configurations. In contrast, the incremental update mechanism is robust enough to cope with any tested burst error conditions at a deadline of 100 ms.

Decreasing the sample deadline for the baseline burst error model shows that the incremental update mechanism in the infrastructure use case also manages to significantly decrease the sample latency as apparent from reliable transmission being possible for all sample deadlines larger than 30 ms. Notably, this is the case despite occasional needs to transmit complete samples with 5% of samples being transmitted completely during this experiment. For longer and more frequent burst errors that deadline does not suffice for reliable transmission any more. For longer and more frequent bursts a sample deadline exceeding 50 and 40 ms is needed, respectively, as the longer/more frequent bursts result in the sample latency being increased as more retransmissions are needed for reliable sample transmission. Nevertheless, regardless of the burst error configuration, the incremental update mechanism allows for a significant reduction in the samples' transmission latency (50–70 ms depending on the burst error configuration).

Concluding the use case, the incremental update mechanism excelled during the burst error stress test as it drastically improves robustness to burst errors and allows for reduction in transmission deadline. The latter directly impacts the total latency of cooperative perception applications relying on that data in an advantageous way. This further allows to drastically decrease the reaction time of perception applications which is a significant QoS gain from an application perspective.

VIII. Physical Proof of Concept

We set up a physical demonstrator for proof of concept experiments evaluating the effectiveness of the proposed mechanisms in the real world. Two x86-nodes running Ubuntu

Fig. 15. Exemplary artifacts occurring when transmitting H.265 encoded streams (DVS dataset [31]) via a wireless connection at a packet loss rate of 0.1% after MAC layer BEC.

20.04 have been used for this purpose. Following [11], 802.11n WiFi expansion cards supporting the ath9k driver, that was deemed most customizable for such a setup, have been used. Given Linux limitations with respect to WiFi configuration, specifically the missing option to set a static modulation and coding scheme (MCS) and thereby fixed data rates, we decided against using an access point (AP) setup and instead opted for an ad-hoc network configuration. However, this limits the setup to the use of 802.11a, with the maximum (though static) data rate, as measured using *iperf*, of 18.5 Mb/s. We acknowledge the difference to the data rates needed to exchange raw sensor data from the two existing datasets (cf. simulation results in Section VII). Nevertheless, scaling the sample size accordingly, we still end up with "large" fragmented samples, that allow to evaluate the effectiveness of the basic concepts proposed in this work.

We begin with an image quality analysis of an H.265 stream (encoded via *ffmpeg*[4]) comprising 1000 consecutive frames from the Digital Vision Security (DVS) dataset [31] transmitted via the wireless channel. As controlling errors in wireless channels is a nontrivial task, we decided to place the two hosts next to each other to minimize fading-related error sources and emulate packet loss using the Linux command *tc*.[5]

Already at a low packet loss rates after the MAC layer BEC of 0.1%, corresponding to a 10% FER and up to 3 MAC layer retransmissions (uniform BER, see Section III), major artifacts occur that obfuscate large portions of the image (Fig. 15). Given realistic burst error characteristics, burst errors exceeding seven consecutive packets—which corresponds to the maximum number of MAC layer retransmissions per packet—have to be expected, with their probability of occurrence exceeding 0.1% (cf. Fig. 14). Hence, even the most robust MAC configuration is not sufficient for eliminating packet loss after the MAC layer BEC. W2RP protection would not help, because of the unavoidable transmission of I-frames becoming the weak link due to the timing issues previously discussed in Section II. Similar errors occur when transmitting A2D2 data [30] recorded by cameras equipped to a driving vehicle.

[4]https://ffmpeg.org/

[5]https://man7.org/linux/man-pages/man8/tc.8.html

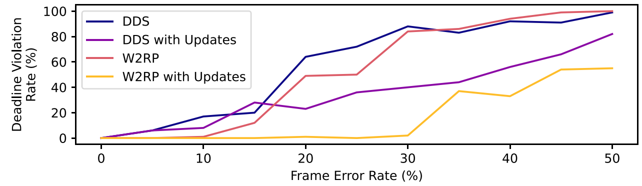


Fig. 16. Robustness of (Fast) DDS and W2RP with and without using the incremental update mechanism for increasing error rates.

This practical result supports the reasoning in Section II, such that we do not pursue the use of video codecs any further.

Next, we investigate the robustness of the incremental update mechanism in a channel affected by transmission errors. Again, hosts are placed next to each other and packet loss is emulated using *tc*. This allows us to evaluate the performance at various operating points (channel conditions). As the basis for our evaluation we used a FastDDS-based W2RP implementation courtesy of [11] and the default FastDDS[6] as provided by eProsima. Both have been modified to allow for dynamic sample sizes and incremental updates according to Section V-A. As the FastDDS version used here is incompatible with the version required by the current Autoware Stack, we had to refrain from also evaluating RoI transmissions.

We exchange 1000 synthetically generated, 60 kB samples (300×200 grayscale images) between the two nodes. The samples are exchanged at a rate of 10 Hz, with the sample deadline being set to 100 ms. The images have been generated in a way that consecutive samples differ by 20%–40% [cf. locality level in Fig. 2(b)]. Both (default Fast)DDS and W2RP have been evaluated with and without using the incremental updates. As apparent from the results in Fig. 16, default DDS only manages to ensure reliable data exchange if no errors occur. Using incremental updates results in lower violation rates for DDS, however, the violation rate still exceeds 0% making resilient application operation impossible. As is, the incremental update mechanism cannot overcome the limitations with respect to the issue of default DDS in wireless channels, e.g., caused by the burst transmission of fragments (cf. [7]). In contrast, even using W2RP for transmitting the complete sample allows for reliable exchange of samples up to an error rate of 10%. Combining W2RP with the incremental update mechanisms shows robustness improvements with reliable exchange feasible up to an error rate of 30%. This confirms the previous results from the simulation. Reducing the effective sample size that needs to be transmitted directly increases slack that results in improved robustness to higher error rates. Other than in case of the H.265 test with standard MAC layer retransmission, all individual sample deadlines are met without any quality degradation.

IX. Conclusion

Current V2X and wireless communication standardization efforts lack support for distributed applications that require

[6]https://github.com/eProsima/Fast-DDS

the timely and safe exchange of large objects. Assuming availability of sufficient channel bandwidth, a main challenge is robustness of such reliable streaming under high FERs. As demonstrated in this article, the use of established video coding, as suggested in V2X roadmaps, is no viable solution for safety-critical perception pipelines, neither with respect to latency nor to reliability. Previous work for the widely used DDS middleware already demonstrated that robustness can be significantly improved by a BEC protocol, W2RP. This article takes a further step exploiting application knowledge for dynamic protocol adaptation with even higher robustness. This article uses two challenging realistic use cases showing that there is no simple solution for all cases, but that different application-specific data dynamics can be covered by two complementary loss-less data reduction methods, namely, using RoI-based communication and incremental updates. The methods were implemented and integrated with existing software frameworks for DDS and automated driving, and evaluated with network simulation and a physical prototype. The evaluation highlighted significant improvements in data quality compared to established video coding as well as reliability and robustness in general when combining the loss-less optimization techniques with efficient BEC (W2RP). W2RP and the complementary pair of data reduction methods are sufficiently general to be used in other applications where DDS is used for communication over standard wireless networks, enabling improved autonomy or remote control for various safety-critical systems, e.g., in the automotive domain or robotics.

References

[1] A. Alalewi, I. Dayoub, and S. Cherkaoui, "On 5G-V2X use cases and enabling technologies: A comprehensive survey," *IEEE Access*, vol. 9, pp. 107710–107737, 2021.

[2] "C-V2X use cases volume II: Examples and service level requirements," 5G Autom. Assoc. (5GAA), Munich, Germany, Oct. 2020.

[3] "Technical specification group services and system aspects; study on enhancement of 3GPP support for 5G V2X services; (Release 16)," 3GPP, Sophia Antipolis, France, Rep. TR 22.886, Dec. 2018.

[4] "V2X sensor-sharing for cooperative and automated driving," SAE Standard J3224, Aug. 2022.

[5] L. Kang, W. Zhao, B. Qi, and S. Banerjee, "Augmenting self-driving with remote control: Challenges and directions," in *Proc. HOTMOBILE*, 2018, pp. 19–24.

[6] S.-C. Lin et al., "The architectural implications of autonomous driving: Constraints and acceleration," in *Proc. ASPLOS*, 2018, pp. 751–766.

[7] J. Peeck et al., "A middleware protocol for time-critical wireless communication of large data samples," in *Proc. IEEE RTSS*, 2021, pp. 1–13.

[8] *IEEE Standard for Information Technology–Telecommunications and Information Exchange Between Systems—Local and Metropolitan Area Networks–Specific Requirements—Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Standard 802.11-2020 (Revision of IEEE Standard 802.11-2016), 2021, pp. 1–4379.

[9] "Technical specification group services and system aspects; release 16 description; summary of Rel-16 work items (Release 16)," 3GPP, Sophia Antipolis, France, Rep. TR 21.916, Sep. 2021.

[10] 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)*, 2019, pp. 1–7.

[11] A. Bendrick et al., "An error protection protocol for the multicast transmission of data samples in V2X applications," *ACM Trans. Cyber-Phys. Syst.*, vol. 18, no. 3, pp. 1–25, 2023.

[12] "Data distribution service, version 1.4," Object Manag. Group (OMG), Needham, MA, USA, document 2015-04-10, Mar. 2015.

[13] (AUTOSAR Org., Mannheim, Germany), *AUTOSAR Specification of Communication Management*. (Nov. 2020). [Online]. Available: https://www.autosar.org/

[14] Y. Lu et al., "Traffic signal detection and classification in street views using an attention model," *Comput. Vis. Media*, vol. 4, pp. 253–266, Aug. 2018.

[15] N. Sperling, A. Bendrick, D. Stöhrmann, and R. Ernst, "Caching in automated data centric vehicles for edge computing scenarios," in *Proc. 60th ACM/IEEE Design Autom. Conf. (DAC)*, 2023, pp. 1–4.

[16] E. S. Lubana and R. P. Dick, "Digital foveation: An energy-aware machine vision framework," *IEEE Trans. Comput.-Aided Design Integr. Circuits Syst.*, vol. 37, no. 11, pp. 2371–2380, Nov. 2018.

[17] G. J. Sullivan, J.-R. Ohm, W.-J. Han, and T. Wiegand, "Overview of the high efficiency video coding (HEVC) standard," *IEEE Trans. Circuits Syst. Video Technol.*, vol. 22, no. 12, pp. 1649-1668, Dec. 2012.

[18] Y. Chen et al., "An overview of core coding tools in the AV1 video codec," in *Proc. Pict. Cod. Symp. (PCS)*, 2018, pp. 41–45.

[19] "Technical specification group services and system aspects; study on scenarios and requirements for next generation access technologies; (Release 17)," 3GPP, Sophia Antipolis, France, Rep. TR 38.913, Mar. 2022.

[20] A. Nasrallah et al., "Ultra-low latency (ULL) networks: The IEEE TSN and IETF DetNet standards and related 5G ULL research," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 1, pp. 88–145, 1st Quart., 2019.

[21] D. Cavalcanti et al., "Wireless TSN: Market expectations capabilities and certification," Avnu Alliance, Beaverton, OR, USA, White Paper, 2022.

[22] A. Mildner, "Time sensitive networking for wireless networks—A state of the art analysis," in *Proc. Semin. IITM*, vol. 33, 2019, pp. 33–37.

[23] T. Striffler, N. Michailow, and M. Bahr, "Time-sensitive networking in 5th generation cellular networks—Current state and open topics," in *Proc. IEEE 2nd 5G World Forum (5GWF)*, Dresden, Germany, 2019, pp. 547–552.

[24] D. Cavalcanti, J. Perez-Ramirez, M. M. Rashid, J. Fang, M. Galeev, and K. B. Stanton, "Extending accurate time distribution and timeliness capabilities over the air to enable future wireless industrial automation systems," *Proc. IEEE*, vol. 107, no. 6, pp. 1132–1152, Jun. 2019.

[25] J. Peeck and R. Ernst, "Enabling multi-link data transmission for collaborative sensing in open road scenarios," in *Proc. 31st RTNS*, 2023, pp. 76–86.

[26] A. Bendrick and R. Ernst, "Hard real-time streaming of large data objects with overlapping backward error correction," in *Proc. 49th IEEE IECON*, 2023, pp. 1–8.

[27] *RTI Connext DDS Core Libraries User's Manual Version 6.1.0*, Real-Time Innov., Inc., Sunnyvale, CA, USA, Apr. 2021.

[28] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell Syst. Tech. J.*, vol. 39, no. 5, pp. 1253–1265, 1960.

[29] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell Syst. Tech. J.*, vol. 42, no. 5, pp. 1977–1997, 1963.

[30] J. Geyer et al. "A2D2: Audi autonomous driving Dataset." 2020. [Online]. Available: https://www.a2d2.audi

[31] B. Simpson and Y. Liu. "Digital vision security (DVS) dataset." 2020. [Online]. Available: https://ziyang.eecs.umich.edu/tools.html

[32] S. Kato et al., "Autoware on board: Enabling autonomous vehicles with embedded systems," in *Proc. ACM/IEEE ICCPS*, 2018, pp. 287–296.

[33] C. Chen et al., "RoIFusion: 3D object detection from LiDAR and vision," *IEEE Access*, vol. 9, pp. 51710–51721, 2021.

[34] "DDS interoperability wire protocol, version 2.3," Object Manag. Group (OMG), Needham, MA, USA, document 2019-04-03, May 2019.

[35] H. Bai and M. Atiquzzaman, "Error modeling schemes for fading channels in wireless communications: A survey," *IEEE Commun. Surveys Tuts.*, vol. 5, no. 2, pp. 2–9, 2nd Quart., 2003.