Reducing Communication Cost and Latency in Autonomous Vehicles with Subscriber-centric Selective Data Distribution

Nora Sperling, Rolf Ernst
Institute of Computer and Network Engineering
Technische Universität Braunschweig
Braunschweig, Germany
{sperling, ernst}@ida.ing.tu-bs.de

***Abstract*—Driving automation has become a major cost factor in automotive design. High computation demand for machine learning (ML) applications and growing sensor resolution and data rate require expensive hardware technology and networking. Newer network technologies and topologies can at best compensate the growing communication demand, but the network still accounts for a complex wiring harness with many dedicated sensor cables. In this paper, we exploit the context specific sensor data access of ML perception applications to minimize the sensor data traffic. For that purpose, we extend the popular Data Distribution Service (DDS) publish-subscribe middleware by a subscriber-centric software caching feature, which is then supported by an appropriate network scheduling. Using realistic data sets and ML applications from the popular Autoware benchmark and a zonal architecture according to the P802.1DG automotive Time-Sensitive Networking (TSN) network profile, we demonstrate that both cabling structure and network latency can be significantly reduced for both 1 Gbps and 10 Gbps TSN technologies. This result enables faster perception and/or lower cable cost, at no loss in data quality or reliability.**

I. Introduction

Autonomous vehicles are equipped with a comprehensive set of high resolution sensors like cameras, LIDAR and radar for in depth and reliable environmental perception, irrespective of weather conditions, as needed in advanced driver assistance systems (ADAS) and autonomous driving (AD). The resulting network data rate and latency are growing with sensor resolution leading to an expensive cabling harness [1]. The ever growing demands in connectivity and bandwidth, together with the objective of reducing the weight and cost of the wiring harness, has resulted in new concepts for in-vehicle networks. Scientific perspective papers on automotive trends [2]–[5], as well as publications from the industry [1], [6] are proposing vehicular electronic/electric architectures (E/E architectures) where sensor data is transmitted to a vehicular central compute unit (VCC), running the vehicle automation functions. Time-Sensitive Networking (TSN), a real-time extension of Ethernet, is the dominant backbone standard for current and future high data rate in-vehicle communication [5], [7], [8]. The standards for automotive Ethernet applications are developed by the working group of the automotive profile P802.1DG for TSN [9]. The use cases for P802.1DG [10] assume a *zonal architecture* approach for future E/E architectures. This architecture concept reduces the complexity of the in-vehicle network by interconnecting topologically separated zones with fewer and shorter wires, using a powerful high bandwidth Ethernet backbone. Notably, the high-resolution sensor streams are excluded from such concentration, as seen in [10], because dedicated cables are needed to match their data rates and latency requirements. The inefficiency of dedicated sensor cables has been recognized, motivating technology around multi-Gigabit Ethernet links like Gigabit Multimedia Serial Link (GMSL) [11], which aim to provide enough bandwidth for multiple infotainment and sensor applications on a single link, enabling a much simpler wiring harness. So far however, new technologies and standards for higher link speed seem to merely compensate the increasing network requirements in data rate and latency that result from the growing time and space resolution of sensor technology. Therefore, harness cost reduction and latency improvement will only be possible with the help of new solutions for data load minimization. Research effort has been spent on the compression and decompression of data. Video codecs like H.264/MPEG-4 Advanced Video Coding (AVC) [12] reduce data size for transport. Yet, video codecs introduce computation latencies into the overall delay of data transfer, as data requires to be compressed and decompressed. Furthermore, the algorithms are not loss free, in particular in rapidly moving scenes of a vehicle in motion. Such losses might affect guarantees of sensor fusion and perception. Data compression algorithms have in common that they exploit data redundancy or relevance to human perception, but are unaware of the relevance of data for subsequent automated driving applications.

Contribution: In this work, we utilize properties of the application, specifically high-resolution perception algorithms, to selectively control data transmission in future vehicle networks. We show that revisiting the established unidirectional publish-subscribe sensor data communication in order to introduce a novel *application aware* subscriber-controlled bidirectional software caching enables a drastically

979-8-3503-8741-4/24/$31.00 ©2024 IEEE

reduced communication latency that can be used to reserve more computation time in tightly constrained end-to-end perception pipelines and/or to share network links for reduced cabling cost.

In the following Section II application-specific data relevancy and timing requirements are highlighted and briefly examined regarding communication. For this purpose, the specific characteristics and challenges of state-of-the-art environmental perception for highly autonomous vehicles will be presented. Further, we will provide a review of related work within the context of in-vehicle communications. The core concept of subscriber-centric selective data communication will be presented in Section III. In Section IV we take a closer look at in-vehicle networking use cases and evaluate the benefits of the proposed communication approach. We conclude this work in Section V.

II. State-of-the-Art and Related Work

A. Application-specific challenges due to big data

Highly autonomous vehicles predominantly utilize machine learning (ML) based detectors for reliable environment perception, especially for image-based sensing. To guarantee safe autonomous vehicle operation in dense traffic, deadlines in end-to-end perception pipelines must be bounded, typically to a maximum of 100 ms [13]. However, the computation time of ML applications strongly grows with the size of their input. With the trend to HD or Full HD camera resolution, computation within 100 ms is a challenge. The authors in [14] have found that object detectors with sufficient accuracy require more than 100 ms to process high resolution input images (1280 × 1920) and even scaled down images (640 × 960) still require at least 60 ms to compute. This barely suffices for the time-critical perception pipeline in autonomous vehicles. The reduced resolution in the input images also affect the detection accuracy, clearly diminishing the benefits of high resolution cameras.

B. Application-specific approach to reduce computation time

To tackle the challenge of long computation times in ML detectors, further concepts to reduce the input data size without scaling down resolutions and reducing quality have been investigated. These concepts revolve around attention boxes and Regions of Interests (RoIs), which are either sub-images that contain relevant information or specific selections. These attention boxes or RoIs are derived from content of current sensor samples, prior samples or further sources of information (cf. in Section III). Lu et al. [15] have shown, that instead of downsizing the original image, cutting an attention box from the original frame increases the target size in the input data and with that, improves the accuracy and inference time of the neural network. But, this concept is not limited to image-based sensing. While LIDAR point clouds hold relevant information on the existence and location of objects, they contain noisy or less useful data as well. The authors in [16] apply a hybrid ground classification along with a RoI identification method to select relevant data from the raw sensor input, improving the overall processing time of the LIDAR data.

C. Exploitability of application-specific data relevance

With growing sensor resolution, not only computation times but also transmission times of data samples from sensors towards the processing unit increase. Even though the approaches under *B.* originally target improvements in computation times, they could be exploited for optimizations of data transmission as well, as only a portion of an image must be present in the memory of the processing node, favoring decentralized data management. However, state-of-the-art vehicle platform designs are locked down on the concept of central data storage. As all sensor data is processed centrally, these platforms feature a data management scheme in which all data is transported to and managed by the VCC. The concept of fully locally available (sensor) data can also be found in software developed for ADAS and AD functionality. Publish-subscribe middlewares are established for inter-module data communication, as seen in the open source AD software stacks from The Autoware Foundation[1], which utilizes DDS [17] and Apollo, using CyberRT[2]. The well-known publish-subscribe middleware DDS has also been adopted by AUTOSAR[3], the global automotive software standard. Hence, the current vehicle data management does not support exploitation of data locality in communication.

D. Data distribution over in-vehicle networks

In [10], the working group of the automotive profile P802.1DG for TSN has defined use cases and requirements for zonal architectures, as well as traffic class assignments for automotive traffic types and network topologies that we refer to in this work. The authors in [18] and [19] simulated TSN-based in-vehicle networks with a zonal architecture in OMNeT++ [20] using sensor stream parameters given by OEMs from Shanghai and Korea, respectively. In TSN-based solutions for in-vehicle networks, sample communications are broken down into application-agnostic fragment-based traffic streams. Consequently, the network evaluations by the authors of [18] and [19] are focused on packet-level latencies and jitter, as it is standard practice for stream-based communication. As a result of the stream-based object transfer, the object transmission latency grows with its update rate, leaving less time for the computation intensive ML applications. In contrast, the evaluations in this paper regarding latencies relate to *sample* transmission latencies, as these are the important metrics to increase the time available for detection within the perception pipeline.

State-of-the-art publish-subscribe middlewares such as DDS, which consider data as object samples and not as packet sets, allow object-specific information such as sample deadlines to be taken into account. But due to the nature of these middlewares that distribute new data samples to every

[1]https://autoware.org
[2]https://developer.apollo.auto/cyber.html
[3]https://www.autosar.org

subscriber at once, the possible improvements in *sensor data transmissions* regarding the RoI optimizations that proved fruitful for the *data processing* steps within the perception flow cannot be applied. DDS [17] distributes samples via the Real-Time Publish-Subscribe (RTPS) protocol. This protocol does not take any form of content feedback from the subscriber to the publisher into consideration. As a result, dynamic application aware data reduction mechanisms prior to transmission are not part of the DDS or RTPS standard. In this work, we base our communication scheme on DDS, as it provides fundamental view on sample characteristics and it is supported by the AUTOSAR standards. However, we extend the DDS/RTPS middleware to be able to handle a feedback-based protocol.

III. Regions of Interest in Sensor Communication

Re-evaluating sensor data communication requires to modify the middleware of an established sensor-actuator software platform. The structure of the traffic light classifier module from Autoware Universe[4] perfectly illustrates the limitations of the publish-subscribe middleware when used for high-resolution input to ML based perception. The module input is a raw camera frame, along with information for relevant RoIs that show traffic lights. Both input data are periodically sent from their respective publishers, the camera and a map-based traffic light detector. Therefore, the first processing step within the classification module is clipping the relevant RoIs from the input camera frame. Only then the actual classification of the traffic light signal follows. In order to be able to make a realistic statement about the size ratio of a raw camera image and the section relevant for traffic light detection, the sizes of the RoIs were recorded during a test run of the Autoware software stack together with the digital twin simulation AWSIM[5], a simulator developed for Autoware development and testing. The RoI information traced on a 100s long drive seen in Figure 1 illustrates that the relevant RoI within a camera frame makes up only a fraction of the original sample. While this relevant region is not known to the camera, the *publisher*, the perception application as a *subscriber* has knowledge on specific RoIs or attention boxes (for example areas around 50% of the frame size). This knowledge is derived either from further contextual information of tracking modules, like the map-based traffic light detector from Autoware, or simply from previous frames. Note that prediction of relevant areas on the basis of previous frames profits from high frame rates. A moving object can be detected with more certainty and robustness, if less time has passed between frames, which allows to narrow down the size of an estimated area of interest, effectively attenuating the required bandwidth and transfer time, when compared to full image transmission.

Figure 2a illustrates the standard procedure of a camera sensor publishing data to a perception node that classifies traffic light states. With standard publish-subscribe sensor

[4]https://autowarefoundation.github.io/autoware.universe/main/

[5]https://tier4.github.io/AWSIM/

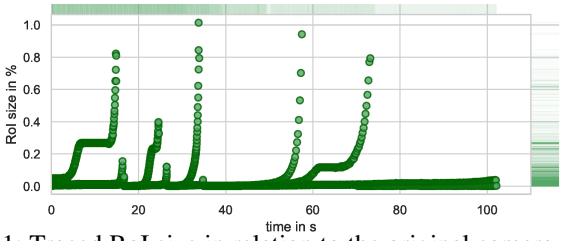


Fig. 1: Traced RoI size in relation to the original camera frame processed by the Autoware Universe traffic light classification module during a 100s simulated drive in AWSIM. Simulated was a driving scene through a model of Nishi-Shinjuku, a district of Shinjuku, Tokyo. The driving scene featured several crossings and turns.

transmission, over 4000 maximum size Ethernet packets are required to transmit a Full HD camera image of about 6,22 MB (cf. Table I), under the assumption that all 1500 bytes of the Ethernet payload can be used for data.

By extending the unidirectional publish-subscribe communication shown in Figure 2a by a *request-reply* or *feedback-based* (terms used interchangeably) communication protocol that enables *caching* of sensor data (see Figure 2b), the data transfer between software modules in general and between VCC and sensors in particular can be improved significantly. While the potentially periodic messages from the publisher towards the subscriber remain, they will not transport sample data, but instead serve as update announcements requiring only one short packet to be sent, see Figure 2b. The subscriber-side middleware responds on reception of such announcement with a specific request for a RoI of dynamic position and size. Transmitting only a single RoI that is 1% of the image size (cf. Figure 1) requires just about 40 Ethernet packets. The RoI request will be handled by the middleware at the publisher. The resulting middleware software changes including memory structure and local caching can be solved in various ways and are beyond this paper. A possible concept is outlined in [21].

The additional computing effort of feedback-based sensor communication from Figure 2b is low. With intelligent list-based software caching, finding the relevant RoI data block in the local memory is in the order of $O(n)$ and, therefore, the cumulative time to select all relevant RoIs from an image will be in the scope of $t_{p,RoI,c}$. Running the Autoware software stack with the AWSIM simulation on a desktop PC (CPU: i7-12700, RAM: 32 GB, GPU: GeForce RTX 4070) with Ubuntu 22.04 has shown that $t_{p,RoI,c}$ is in the range of 70 to 700 µs. As request-reply RoIs transmissions are made of only single packets or small bursts of packets, depending on the size of the RoI, t_{net1}, t_{net2} and t_{net3} are all significantly smaller than t_{net} from Figure 2a, so are the instances of $t_{com,f}$ compared to $t_{com,s}$. These transport delays and communication stack delays of single packets or small bursts of packets make up the further delay until the classifier on the perception processing unit can classify the RoIs. It is to note, that $t_{p,RoI,d}$ can be disregarded, as RoIs are determined by an external module, independent of the use of the sensor data transport mechanism proposed in this paper, and are available with reception of the

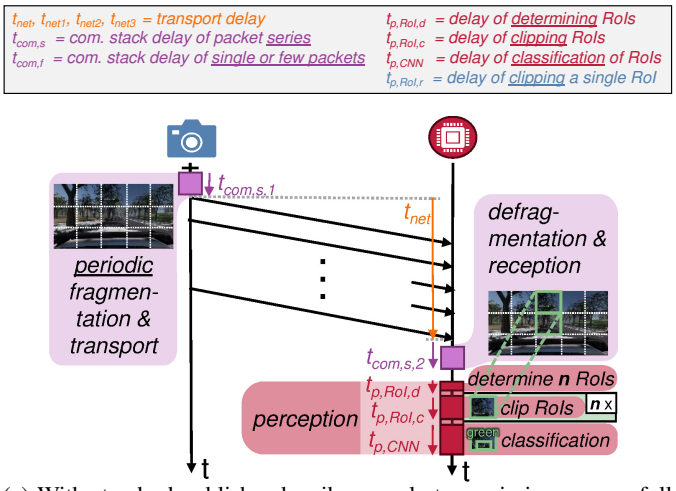
camera image, as described at the start of this Section III. Therefore, the added communication delays of feedback-based sensor communication is determined by the round-trip time (RTT) of the protocol messages. Overall, the reduction of data by utilizing request-reply communication allows to a) drastically reduce transmission latencies and b) results in lower communication resource utilization.

In this paper, we focus on the network traffic produced by such a request-reply protocol and the unique characteristics of its bidirectional request and reply messages, whereby replies are sporadic and dynamic in size. This is very different from the established publish-subscribe traffic. In Section IV, we evaluate the protocol, comparing the transport delays of the small, dynamic RoIs with the additional RTT from Figure 2b to the transport delay in Figure 2a. We utilize an OMNeT++ [20] network simulation, which is sufficient to investigate the impact on network load and timing.

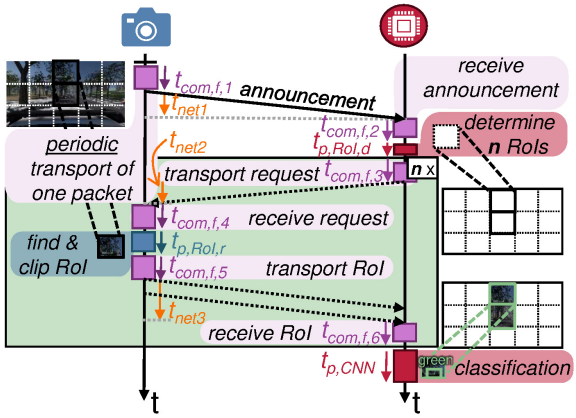
IV. Networking Use Cases with RoIs

Vehicles at higher levels of automation are equipped with several high-bandwidth sensors consisting of typically 8 or more cameras (see Figure 3a) and 4 to 5 zonal LIDAR's and radars [23], [24]. In the use cases for the automotive TSN profile P802.1DG [10], a zonal in-vehicle network architecture utilizes a powerful Ethernet backbone to connect the four (or more) vehicular zones. However, regardless of the zonal locality of sensor nodes, the TSN automotive profile suggests point-to-point links for the high-bandwidth sensors, see Figure 3b. Environmental data produced by the surround sensors is processed at the VCC in the context of ADAS or AD functionality, like environment perception for assisted maneuvering and emergency interventions. Due to the central processing of sensor data and the resulting additional point-to-point sensor links, the complexity of the interface in the VCC is aggravated. For 8 cameras and 4 LIDARs, 12 dedicated links with connectors and lengths of one to several meters each are required, even more in case of redundant network design. Overall, weight and complexity of the wiring harness will remain a challenge unless link sharing between high-bandwidth sensors is possible. The arguments in favor of point-to-point connections are based on the required bandwidth in combination with the nevertheless short latency requirements of the data. LIDAR sensors require a bandwidth of more than 100 Mbps individually, while transmitting raw camera images with a Full HD resolution every 100 ms requires a bandwidth of around 500 Mbps, see Table I. Further increasing the image resolution and/or frame rate drives the required bandwidth into the range of multiple Gbps per camera. Alternatively, transferring only relevant RoIs of sensor objects like traffic light details from camera images, allows to reduce the object size to less than 1% of the original sample, as seen in Figure 1. In the following, the effects and improvements that feedback-based RoI transfer provides will be elaborated.

We focus on two goals *(A) reducing sensor data latencies for faster perception* and *(B) enabling link sharing.* All evaluations were made in regard to these conflicting goals.



(a) With standard publish-subscribe sample transmissions, every full camera image is fragmented into a series of packets, which are transmitted one by one. The application can then access the image to process the n RoIs in the image with the delay of $t_{com,s,1} + t_{net} + t_{com,s,2}$.



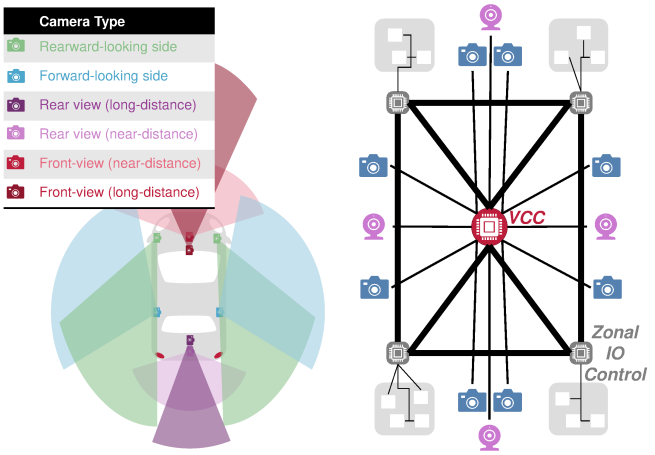
(b) With request-reply transmission of partial samples, subscribers are notified of new samples. A subscribing application then makes a request for a RoI they require, determined with the mechanisms described in Section III. The requested partial data sample is transmitted in burst fashion to the subscriber. A request is triggered and must be handled n times, for every of the n RoIs in the image.

Fig. 2: Comparison of (a) the standard unidirectional publish-subscribe protocol and (b) the modified request-reply communication. The processing time within the perception module is made up of the time to determine the RoIs $t_{p,RoI,d}$, the cumulative time to cut out the n RoIs from the camera image $t_{p,RoI,c}$ and the inference time of the traffic lights in the n RoIs $t_{p,CNN}$.

The elaboration is supported by simulated results of the corresponding traffic configurations found in Table I and the recorded RoI trace from Autoware Universe in Figure 1. The in-vehicle network was simulated with an openly available TSN simulator [25] that is based on OMNeT++. With regard to the necessary reduction of sample latencies, we utilize burst transfer of samples in this paper as already mentioned in Section II. The reference parameters for the LIDAR stream from [18] were adapted accordingly to implement sample burst traffic. The sample burst communication of the data

TABLE I: Traffic parameters of sensors and automotive streams like critical and non-critical control streams and infotainment that flood towards the VCC. *Note: Traffic marked with * was originally stream-based, but was transformed into periodic burst transfers*

				Camera Configurations			
	Control [22]	*LIDAR** [18]	*Radar* [18]	*Full HD Camera*	*RoI Camera* [21]	*Body ECU* [18]	*Audio* [18]
message size	230 B	125 KB	42 B	6,22 MB	<100 KB	64 B	234 B
sampling period	500 µs	10 ms	1000 µs	100 ms	100 ms	100 µs	200 µs
bandwidth	~3.5 Mbit/s	100 Mbit/s	~0.5 Mbit/s	~500 Mbit/s	~8 Mbit/s	~5 Mbits/s	~10 Mbit/s
sample deadline	100 µs	10 ms	1000 µs	100 ms	100 ms	*non timing critical data*	
traffic priority	7	6	6	6	6	3	4



(a) Camera setup for AD or extensive ADAS.

(b) Zonal in-vehicle architecture with privately connected sensors.

Fig. 3: Camera setup of vehicles with high automation level and their suggested underlying vehicle platform network found in automotive TSN profile [10].

samples was simulated using an implementation of the RTPS protocol, whereas the request-reply RoI communication seen in Figure 2b was realized through an extension of this RTPS implementation. We evaluate the effects of inter-stream interference on a link shared by two cameras and a LIDAR sensor. This sensor combination is based on the zonal sensor configuration from Figure 3b. Due to the low resource demands of radar streams, we have omitted them from our shared link evaluations. In this paper, we will not exploit scheduling, but assume conservative worst cases of burst scheduling. In fact, our evaluations will show that complex scheduling techniques become less relevant with RoI transmissions.

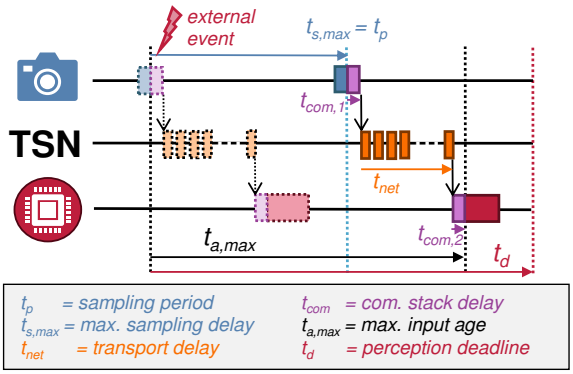


Fig. 4: The max. input age $t_{a,max}$ is composed of the max. sampling delay $t_{s,\,max}$, the transport delay t_{net} and communication stack delay t_{com}. Further camera latencies are negligible.

A. Reducing Sample Latencies

To be able to detect an external event within the required 100 ms, the maximum data age of the input data of perception modules must be clearly below this threshold to allow for enough processing time and a timely reaction. For a precise two-stage detector with the inference time of 60 ms on a full low resolution image from [14], the max. data age $t_{a,max}$ of the camera image would have to remain under 40 ms. The maximum data age of the perception input is composed of the following delays (cp. Figure 4):

$$t_{\mathrm{a,max}} = t_{\mathrm{s,max}} + t_{\mathrm{net}} + t_{\mathrm{com}} \tag{1}$$

Assuming a camera frame rate of 30 frames per second (FPS), a maximum sampling latency (from the occurrence of an external event to its first capture) of 33 ms will be observed as base delay $t_{s,\,max}$. The total communication stack delay t_{com} along with the transmission delay t_{net} of the sample that is typically transmitted in fragments, see Figure 4, must be added to this base delay. Hence, the deadline assumptions in Table I might suffice in slow moving vehicles, but will not suffice in real traffic, as the sampling delay $t_{s,max}$ already meets the perception deadline. Even a frame rate of 20 FPS with a sampling latency of 50 ms would not allow for the estimated computation time of 60 ms. Therefore, a frame rate of 30 FPS with a sampling latency of 33 ms can be regarded as a lower frame rate limit for safe maneuvering in real traffic.

The authors of [26], [27] and [28] have evaluated that a delay introduced by packet processing via a standard or an optimized Linux kernel is in the scope of several hundred µs up to a few ms, depending on the background and peak load. For example, the authors in [28] have measured Linux kernel latencies of up to 770 µs on an ARMv7 architecture. We have therefore set an upper limit of 1 ms on this latency. Using these data, the overall network stack delay t_{com} for a complete sample transmission has an estimated upper bound of 2 ms (1 ms each at egress and ingress, $t_{com,s,1}$ and $t_{com,s,2}$ in Figure 2a). In case of feedback-based RoI transmission, the total communication stack delay for all RoIs has an estimated upper bound of 4 ms. As the short requests and replies can be processed in a fraction of $t_{com,s}$, the latencies $t_{com,f,1}$, $t_{com,f,2}$, $t_{com,f,3}$, $t_{com,f,4}$ were estimated with half a millisecond each. $t_{com,f,5}$ and $t_{com,f,6}$ are dependent on the size of the RoI, and were conservatively upper-bounded with 1 ms, similar to $t_{com,s}$. In Table II the max. data input age that is depicted in Figure 4 is calculated with the estimated network stack delay for four different camera sensor and link configurations, that were simulated with the OMNeT++ TSN simulator mentioned above. The simulated data streams in the OMNeT++ TSN

TABLE II: Camera data input age $t_{a,max}$ [in ms] for different sensor configurations calculated with (1) and simulated sample transmission delays t_{net} [in ms] on shared and dedicated 1 and 10 Gbps links and an estimated communication stack delay t_{com} of 2 ms for complete sample communication and 4 ms for RoI communication respectively. Configurations in which the data rate of the sensors surpass the bandwidth of the link have been marked with *overload*.

			1 Gbps				*10 Gbps*				
	Link	*Load*	t_{net}	$t_{a,max}$ (20FPS)	$t_{a,max}$ (30FPS)	$t_{a,max}$ (50FPS)	t_{net}	$t_{a,max}$ (20FPS)	$t_{a,max}$ (30FPS)	$t_{a,max}$ (50FPS)	$t_{a,max}$ (100FPS)
A	dedicated	1x 6,22MB Image Burst	59.71	*overload*	*overload*	*overload*	5.97	57.97	40.97	27.97	17.97
B	shared	2x 6,22MB Image Bursts	*overload*	*overload*	*overload*	*overload*	11.15	63.15	46.15	31.15	*overload*
C	shared	2x <62KB RoI (<1%) Bursts	0.324	54.324	37.324	24.324	0.029	54.029	37.029	24.029	14.029
D	shared	2.76MB RoI (50%) + <62KB RoI (<1%) Bursts	30.49	84.49	67.49	*overload*	2.74	56.74	39.74	26.74	16.74

simulator are based on the Full HD camera configuration in Table I and a realistic RoI recording from AWSIM (cp. Figure 1).

In case of direct burst sample transfer, the required transmission time for full sensor data has grown rapidly with the growth of sensor data samples. Use case *A* in Table II shows the simulated camera sample transport delays t_{net} and their respective calculated maximum data age after reception $t_{a,\,max}$ for a camera that is traditionally connected via a dedicated link, like seen in Figure 3b. It can be seen, that the burst transmission of a raw Full HD camera sample via a 1 Gbps link already requires two thirds of the 100 ms budget from the perception pipeline without including any sampling delay. As a result, a feedback oriented communication concept that allows to reduce the amount of sensor data to be sent is required to improve the transmission time of relevant sample data. Use cases *B*, *C* and *D* in Table II show the camera sample transport delays t_{net} and resulting maximum data ages $t_{a,\,max}$ for a sensor group consisting of two Full HD cameras and a LIDAR sensor on a shared link with the different combinations of camera sample transmission schemes. Camera sensor configurations that allow for a maximum camera input age $t_{a,\,max}$ within 40 ms and therefore leave enough room for the referenced computation of 60 ms, are highlighted in green. Additionally, camera configurations that are not feasible as data rates surpass the link bandwidth, are highlighted in red and marked with *overload*. From the transport delays t_{net} from use case *C* it can be seen that the required RoIs from the Autoware RoI trace can be requested and transmitted in a fraction of the raw sample's complete transmission time (see Table II). While use case *C* is a rather extreme example as not every perception module might be able to reduce relevant data regions to a 62 KB RoI (about 1% of a Full HD sample), use case *D* demonstrates that reducing only one camera stream to specific RoIs and transmitting rough attention areas with the second camera (50% RoI in Table II) already brings several improvements. Use cases *C* and *D* show that by utilizing selective request-reply data transmissions to greatly decrease the overall data load on the shared link allows to increase sampling rates and as a result, reduce the sample age at the perception's input. If RoI selections are based on previous samples, higher sample rates allow for a more robust and precise RoI prediction, which on the other hand also decreases required RoI sizes. In case RoI sizes amount to full image sizes, timing of use case B is the worst case temporary overload of use case D. However in edge cases, a support for transient reduction of the frame rate would allow compensation of extreme RoI coincidence avoiding overload situations.

B. Cost Reduction through Link Sharing

The stream parameters in Table I assume merely a maximum resolution of Full HD and a frame rate of 10 FPS, which is not sufficient for the requested $t_{a,max}$ of 100 ms. It becomes apparent that camera sensors cause the majority of the overall network utilization. Link sharing within the sensor group at the front of the vehicle in Figure 3b would require 10 Gbps to meet a $t_{a,max}$ of 40 ms. Reducing sample sizes to the size of a RoI of about 62 KB does not only allow to supply multiple Full HD camera sensors with one 1 Gbps link. It further allows to increase the update rate and consequently improve the refresh rate of the perception pipeline, while even reducing the overall bandwidth required.

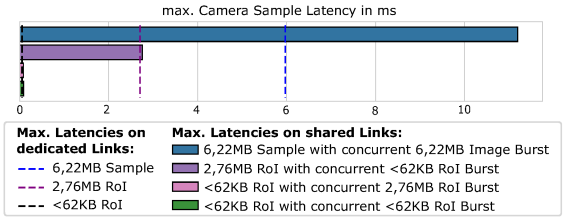


Fig. 5: Camera sample transmission latencies on a 10 Gbps link shared between two cameras. The dashed lines depict the respective transmission times on dedicated, non-shared link.

TABLE III: Mean sample transport delays t_{net} of LIDAR sample bursts on a shared 1 and 10 Gbps links.

	Link	t_{net} (1 Gbps)	t_{net} (10 Gbps)
A	dedicated, non-shared link	1.19 ms	119.96 µs
B	shared, 2x 6,22MB Image Bursts	*overload*	396.43 µs
C	shared, 2x <62KB RoI (<1%) Bursts	1.2104 ms	121.13 µs
D	shared, <62KB (<1%) + 2,76MB RoI Bursts (50%)	1.2145 ms	121.13 µs

In case these high-bandwidth sensors share links and no further prioritization on packet level is available, transmissions of large data samples are subject to extreme inter-stream interference. Figure 5 shows the transmission latencies of camera samples on a shared link of non-coordinated, therefore possibly simultaneous, camera and LIDAR sample bursts with

the configurations seen in Table I and the recorded RoI trace as input for the RoI burst traffic. The transmission time for the single Full HD camera sample, which would require only about 6 ms on a dedicated link, is almost doubled, delayed primarily by its counterpart on the shared link. The time required to transmit two concurrent RoIs is only a fraction of the full sample transmissions. Additionally, only for RoI communication on a shared link, the transmission times are in the same scale as the sample transmission times expected on a dedicated link. A similar trend can be seen for LIDAR sample transmissions in Table III. As could be seen with the camera sample transmissions, when transmitting only requested RoIs of camera samples, LIDAR sample transmissions require only as much time on a shared link, as they would on a private link. It is to note, that these sample latencies solely include transmission times of the protocol messages (namely sample announcement, RoI request and RoI reply) and they do not include the estimated 4 ms network stack delay.

V. Conclusion

In future autonomous vehicles, a large number of high resolution sensors will feed their data into a high-performance vehicular central compute unit. The data transfer does not only generate high network load, but is also responsible for long communication times that add to the sampling delay and cut the time that is available for computing the perception network. Increasing the sampling rate to reduce the sensor sampling delay is in conflict with the resulting higher network load. In consequence, even a 10 Gbps TSN network has limited link sharing capacity. However, if the perception application can adaptively select relevant regions of interest in a 2-stage approach, then we can enhance the automotive middleware by a subscriber-centric software cache protocol. This protocol can drastically reduce latency and network load thereby giving room for higher sensor frame rates that significantly reduce data age and perception delay and/or allow for more time to compute the perception network. Utilizing feedback-based communications supports dynamic system adaptation without causing great impact on the resulting traffic load. The overall lower bandwidth requirements also permit sensor stream integration, with the option of reducing the expensive wiring harness. The evaluation with OMNET++ using available data and perception applications from the AD software stack Autoware and a zonal architecture, as proposed in the automotive TSN profile, demonstrates huge performance and cost cutting potential. We also explained how the potential risk of losing data in a transient network overload can be mitigated with adaptive frame sampling. This result should be an incentive for further research into perception applications that use context information for input data selection.

VI. Acknowledgments

This work was supported by the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation) under Grant ER168/32-2 and by the German BMBF under FKZ 01IS22088 (AUTOTECHagil).

References

[1] TOSHIBA ELECTRONICS EUROPE GMBH, "Will Automotive Ethernet put vehicle cable put vehicle cable harnesses on a much needed diet?" Accessed: 2024-03-6. [Online]. Available: https://toshiba.semicon-storage.com/content/dam/toshiba-ss-v3/master/en/semiconductor/design-development/innovationcentre/whitepapers/TCM0450A_ENG.pdf

[2] A. Bucaioni and P. Pelliccione, "Technical architectures for automotive systems," *2020 IEEE International Conference on Software Architecture (ICSA)*, 2020.

[3] O. Alparslan *et al.*, "Next generation intra-vehicle backbone network architectures," in *IEEE 22nd International Conference on High Performance Switching and Routing (HPSR)*, 2021.

[4] M. Z. Bjelica *et al.*, "Central vehicle computer design: Software taking over," *IEEE Consumer Electronics Magazine*, vol. 8, no. 6, 2019.

[5] L. L. Bello, *et al.*, "Recent advances and trends in on-board embedded and networked automotive systems," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 2, 2019.

[6] V. Navale *et al.*, "(R)evolution of E/E architectures," *SAE International Journal of Passenger Cars - Electronic and Electrical Systems*, 2015.

[7] S. Brunner *et al.*, "Automotive e/e-architecture enhancements by usage of ethernet tsn," in *13th Workshop on Intelligent Solutions in Embedded Systems (WISES)*, 2017.

[8] L. Lo Bello *et al.*, "A perspective on ethernet in automotive communications—current status and future trends," *Applied Sciences*, 2023.

[9] "P802.1DG – TSN Profile for Automotive In-Vehicle Ethernet Communications," *IEEE Standard 802.1DG*, 2023.

[10] D. Pannell *et al.*, "Use Cases – IEEE P802.1DG," Time-Sensitive Networking Task Group, Tech. Rep., Jul. 2019. [Online]. Available: https://www.ieee802.org/1/files/public/docs2019/dg-pannell-automotive-use-cases-0919-v04.pdf

[11] G. Ostrem, "GMSL: Gigabit Multimedia Serial Link – An Introduction," in *Handbook of Visual Display Technology*, 2020.

[12] C. ITU Telecom, "Advanced video coding for generic audio-visual services," 2003.

[13] S.-C. Lin *et al.*, "The architectural implications of autonomous driving: Constraints and acceleration," *SIGPLAN Not.*, 2018.

[14] M. Carranza-García *et al.*, "On the performance of one-stage and two-stage object detectors in autonomous vehicles using camera data," *Remote Sensing*, 2021.

[15] Y. Lu *et al.*, "Traffic signal detection and classification in street views using an attention model," *Computational Visual Media*, 2018.

[16] J. Choi *et al.*, "Multi-target tracking using a 3d-lidar sensor for autonomous vehicles," in *ITSC 2013*, 2013.

[17] Object Management Group (OMG), "OMG DDS (Data Distribution Service) Standard," 2015, Accessed: 2023-03-15. [Online]. Available: https://www.omg.org/spec/DDS/1.4/PDF

[18] F. Luo *et al.*, "Design methodology of automotive time-sensitive network system based on omnet++ simulation system," *Sensors*, 2022.

[19] J. Lee *et al.*, "Time-sensitive network (tsn) experiment in sensor-based integrated environment for autonomous driving," *Sensors*, 2019.

[20] A. Varga, "OMNeT++," in *Modeling and Tools for Network Simulation*, Berlin, Heidelberg, 2010.

[21] N. Sperling *et al.*, "Invited: Caching in automated data centric vehicles for edge computing scenarios," in *DAC 2023*, 2023.

[22] C. Park *et al.*, "Performance evaluation of zone-based in-vehicle network architecture for autonomous vehicles," *Sensors*, 2023.

[23] A. Rangesh *et al.*, "A multimodal, full-surround vehicular testbed for naturalistic studies and benchmarking: Design, calibration and deployment," 2019.

[24] C. Wang *et al.*, "On the application of cameras used in autonomous vehicles," *Archives of Computational Methods in Engineering*, 2022.

[25] Computer Institute of and Engineering Network, IDA, "IDA TSN Simulator," 2020. [Online]. Available: https://github.com/IDA-TUBS/IDATSNSimulator

[26] K.-B. Gemlau *et al.*, "Efficient timing isolation for mixed-criticality communication stacks in performance architectures," in *ETFA 2022*, 2022.

[27] L. Abeni *et al.*, "Investigating the network performance of a real-time Linux Kernel," in *Proc. 15th Real Time Linux Workshop*, 2013.

[28] C. S. V. Gutiérrez *et al.*, "Real-time Linux communications: an evaluation of the Linux communication stack for real-time robotic applications," *arXiv preprint arXiv:1808.10821*, 2018.