Received October 16, 2020, accepted November 24, 2020, date of publication December 1, 2020,
date of current version December 10, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3041688

On the Potential of V2X Message Compression for Vehicular Networks

MIGUEL SEPULCRE[1], (Senior Member, IEEE), JAVIER GOZALVEZ[1], (Senior Member, IEEE), GOKULNATH THANDAVARAYAN[1], (Graduate Student Member, IEEE), BALDOMERO COLL-PERALES[1], (Member, IEEE), JULIAN SCHINDLER[2], AND MICHELE RONDINONE[3]

[1]Communications Engineering Department, Universidad Miguel Hernández de Elche (UMH), 03202 Elche, Spain
[2]German Aerospace Center (DLR), Institute for Transportation Systems, 38104 Braunschweig, Germany
[3]Hyundai Motor Europe Technical Center, 65428 Rüsselsheim, Germany

Corresponding author: Miguel Sepulcre (msepulcre@umh.es)

This work was supported in part by the Horizon2020 Programme through the TransAID Project under Grant 723390, and in part by the Spanish Ministry of Science and Innovation (MCI), the State Research Agency (AEI), and the European Regional Development Fund (FEDER), under Project TEC2017-88612-R.

ABSTRACT The emergence of connected automated vehicles and advanced V2X applications and services can challenge the scalability of vehicular networks in the future. This challenge requires solutions to reduce and control the communication channel load beyond the traditional congestion control protocols proposed to date. In this paper, we propose and evaluate the use of V2X message compression to reduce the channel load and improve the scalability and reliability of future vehicular networks. Data compression has the potential to reduce the channel load consumed by each vehicle without reducing the amount of information transmitted. To analyze its potential, this paper evaluates the compression gain of three compression algorithms using standardized V2X messages for basic awareness (CAMs), cooperative perception (CPMs) and maneuver coordination (MCMs) extracted from standard-compliant prototypes. We demonstrate through network simulations that V2X message compression can reduce the channel load. In particular, the tested compression algorithms can reduce the channel load by up to 27% without reducing the amount of information transmitted. Reducing the channel load and the consequent interferences significantly improves the reliability of V2X communications. However, this study also emphasizes the need for high-speed compression and decompression modules capable to compress and decompress V2X messages in real time, especially under highly loaded scenarios.

INDEX TERMS Vehicular networks, V2X, message compression, congestion control, channel load, interference, reliability, scalability, CAM, CPM, MCM.

I. INTRODUCTION

Vehicular networks enable the continuous exchange of information between vehicles and other nodes using V2X (Vehicle-to-Everything) communications. The emergence of connected automated vehicles will increase the demand for advanced V2X applications and services and therefore the information exchanged. This can challenge the scalability of V2X networks in the next years. To address this challenge, it is necessary to reduce and control the communication channel load and the interferences present in the network. This has been traditionally addressed through congestion control, and different congestion control protocols have been proposed to date. These protocols normally adapt the communication parameters, e.g. the message rate [1], [2], the transmission power [3], [4], or the data rate [5], [6] to control the load. These solutions reduce the channel load at the expense of modifying the amount of information exchanged, the communication range or the protection against transmission errors [7]. These changes can negatively affect the execution of applications [8] and alternative approaches should be considered. An alternative approach to reduce the channel load and improve the scalability of vehicular networks is data compression. Data compression can reduce the amount of

The associate editor coordinating the review of this manuscript and approving it for publication was Xijun Wang.

bits per message without affecting the information effectively transmitted per vehicle nor modifying the communication parameters. Data compression is widely used in communication systems to improve the bandwidth utilization. For example, HTTP compression can be applied in web servers and web clients (browsers) to improve transfer speed and bandwidth utilization [9]. HTTP compression for text files can achieve compression gains of around 75% [10]. However, the use of data compression in vehicular networks has been underexplored. In this paper, we propose and evaluate the potential of data compression to reduce the channel load and improve the scalability of vehicular networks without modifying the information transmitted or the message rate. To make it agnostic from the radio access technology, we propose that the V2X messages generated by the upper layers of the protocol stack are compressed before they are sent down to the Transport & Network layer. A V2X message compression component could be implemented at the Facilities layer of the ETSI [11] or ISO [12] ITS Architectures (see Figure 1), or above the WSMP Transport Layer of the 1609/WAVE Architecture [13]. This component would be in charge of the message compression at the transmitter and the decompression at the receiver. It would also be independent of the radio access technology considered. The use of data compression would not require any other significant modification to the protocol stack, which increases its potential for future standardization.

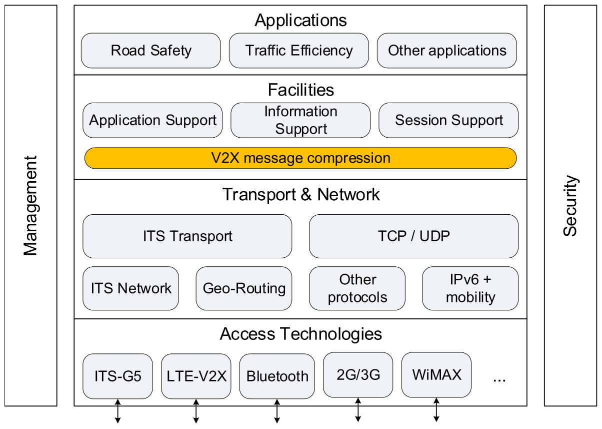


FIGURE 1. ETSI ITS architecture with proposed V2X message compression.

The compression gain significantly depends on the size and statistical properties of the input data. Previous studies have shown that text files can be significantly compressed because they have repeated substrings (e.g. words) [14]. However, V2X messages have a relatively small size (hundreds of Bytes) and their properties and potential for data compression have not been studied yet. In this paper, we propose and evaluate the use of data compression using standardized V2X messages obtained from experimental prototypes. We consider three different compression algorithms and compare the compression they can achieve. In addition, we evaluate by means of network simulations the positive effects of the V2X message compression on the channel load and V2X communications reliability. To conduct this study, we apply data compression to different V2X messages standardized by ETSI: CAMs (Cooperative Awareness Messages), CPMs (Collective Perception Messages) and MCMs (Maneuver Coordination Messages). These messages are coded in prototypes that implement the ETSI ITS architecture and use ITS-G5 (adaptation of IEEE 802.11p) for the physical and MAC (Medium Access Control) layers. Experimental data to build and transmit the V2X messages has been obtained from field trials. The prototypes and field trials have been implemented in the H2020 TransAID project [15]. The use of standard-compliant prototypes and standardized V2X messages is important because the properties of the data to be compressed can influence the compression gain that can be achieved.

This paper is structured as follows. Section II presents the compression algorithms considered in this study. Section III describes the V2X messages that have been employed. Section IV presents the experimental testbeds that have been used for the collection of the V2X messages. Section V analyzes the statistical properties of the V2X messages collected, and evaluates the compression gain of the different algorithms. It also compares the compression gains achieved with the best algorithm to its theoretical compression limit, and measures the time needed to compress and decompress V2X messages. Section VI evaluates the impact of message compression on vehicular networks by means of simulation. Section VII presents the main conclusions achieved.

II. DATA COMPRESSION

Data compression is the process of encoding information to reduce the number of bits of the original representation. It is also known as source coding in data transmission. Data compression can be either lossy or lossless. Lossy compression reduces the number of bits by removing unnecessary or less important information, and is typically applied in audio or image compression. Lossless compression reduces bits by identifying and eliminating statistical redundancy. Lossless compression can perfectly reconstruct the original data from the compressed one without any loss. Considering the critical nature of V2X information, this work focuses on lossless compression and we study two types of compression methods: entropy compression and adaptive dictionary compression [16]. Adaptive dictionary compression is one of the most common methods. Different algorithms of this type have been shown to be computationally efficient and universal, i.e. they do not require prior knowledge of the data to be compressed. As a consequence, this method can be applied to any V2X message without any previous analysis or processing. Its main drawback for the compression of V2X messages is that it only becomes effective for long bit sequences and therefore its compression gain for relatively small messages needs to be studied. Entropy compression is also widely used in many different fields because of their simplicity, and lack of patent coverage. In contrast to adaptive dictionary compression, entropy compression is not a universal compression method. Entropy compression thus requires prior knowledge of the data to be compressed. To apply it in vehicular networks, the statistical properties of the V2X messages need to be analyzed to build a static dictionary, agreed by all vehicular nodes. Its main advantage is that, once this dictionary is built, its compression gain does not depend on the message length.

A. ENTROPY COMPRESSION

The idea behind entropy compression (also known as probability compression) is to divide the message source in symbols of equal length. The alphabet is the set of possible symbols. Frequent symbols are encoded in fewer bits than infrequent symbols. Therefore, compression algorithms that make use of entropy compression take into account the probability that each symbol appears in a message. Using these probabilities, a table of codewords or dictionary is constructed. Codewords for symbols with low probabilities have more bits, and codewords for symbols with high probabilities have fewer bits so that the input data can be effectively compressed.

Different algorithms have been proposed in the literature to construct the table of codeworks or dictionary. One of the most used algorithms is Shannon-Fano coding [18]. It is able to derive compact dictionaries, i.e. with the shortest average codeword length to represent the messages of a given source. Assuming that the probability of each symbol in the alphabet is known, the Shannon-Fano coding algorithm operates as follows:

1. Sort the list of symbols in decreasing order of probability, the most probable ones to the left and least probable to the right.
2. Split the list into two parts with the total probability of both parts being as close as possible.
3. Assign bit 0 to the left part and bit 1 to the right part.
4. Repeat the steps 2 and 3 for each part until all the parts are split into individual symbols.

This algorithm is used to build the dictionary of codewords. It only needs to be executed if the probabilities of the symbols of the alphabet change. Once the dictionary of codewords is obtained, it is used to compress the messages by replacing each symbol with its codeword. It is also used to decompress the messages by replacing each codeword with its corresponding symbol.

To compute the probability of each symbol, two different approaches could be envisaged for V2X messages:

Approach 1: Compute the probabilities of the different symbols for each V2X message individually before it is transmitted. This approach implies a different dictionary of codewords for each V2X message. It ensures that the dictionary constructed for each message is the optimal one. However, the receiver needs the dictionary for decompressing the message and the transmitter must send it to the receiver together with the message. This can reduce the gains achieved with message compression since V2X messages are generally of small size.

Approach 2: Compute the probabilities for a set of V2X messages of certain type (e.g. CAM, CPM or MCM) to construct one dictionary per V2X message type. The dictionary does not need to be transmitted with each message, if it is fixed and known by the transmitter and receiver. The main drawback of this approach is that the dictionary is the optimum for the set of messages analyzed, but might not be optimal for each individual message. However, we adopt this approach in this study since the need to append the dictionary to each message in the first approach significantly reduces the gains of compression.

B. ADAPTIVE DICTIONARY COMPRESSION

Adaptive dictionary compression method does not need to parse data before compressing to calculate the symbols' probabilities. It compresses messages by looking for repeated substrings in the input data. At the start, compression algorithms that use this method have no dictionary or use a default baseline one. As compression proceeds, the algorithms add new symbols to the dictionary following certain rules. The algorithms read the input data and search groups of symbols that appear in the dictionary. If a string match is found, a pointer or index into the dictionary is sent to the output instead of the symbol. The compression ratio improves with longer matches. Algorithms based on adaptive dictionary compression have become the de facto standard for general-purpose data compression due to their high-performance compression combined with reasonable memory requirements [16]. One of the most widely used algorithms based on adaptive dictionary compression is the so-called Lempel-Ziv algorithm. It is a universal algorithm that has been shown to be computationally efficient. It can therefore be applied to any V2X message without any previous analysis, but its compression gain could be limited when the amount of data to be compressed is small. In this study, we have used two open-source tools that implement this algorithm: Compress [19] and Gzip [20].

Gzip is based on the LZ77 algorithm [21], which is the original algorithm proposed by Lempel and Ziv. LZ77 makes use of a sliding window buffer to look for repeated substrings in the input data. The sliding window buffer is divided in two parts: a search buffer and a lookahead buffer. The search buffer contains the data that has already been compressed. The lookahead buffer contains the data that has not been compressed yet. As data is compressed, the oldest compressed data is removed from the search buffer and new uncompressed data is added to the lookahead buffer. Data is compressed when a substring in the lookahead buffer is found in the search buffer. When this happens, the substring is replaced by a pointer that contains its position and length within the search buffer. The longer the substrings, the higher the compression gain. The Gzip output format is described in RFC 1952 [22]. It contains the compressed data together with a series of additional headers. These headers occupy at least 18 bytes and contain a CRC-32 checksum and the length of the original uncompressed data, among other information.

Compress is based on an evolution of the Lempel-Ziv algorithm that is known as LZW [23]. LZW uses the input data to construct a dictionary. The algorithm is normally initialized with 256 entries, each of them one-byte long. The input data is then parsed looking for substrings that appear in the dictionary. When a substring S of the input data is found in the dictionary, it is substituted by its index and a new entry is added to the dictionary. This new entry contains S and the next symbol in the input data. A new entry is therefore added if the dictionary contains a prefix one byte shorter (e.g. "car" is only added to the dictionary if "ca" has appeared in the data).

In general, the size of the input data and the distribution of common substrings significantly affect the compression gain of compression algorithms based on adaptive dictionary like LZ77 (Gzip) or LZW (Compress). Using text as input data, LZ77 is able to produce a compression gain of 60-70% and LZW around 50-60% [19], [20]. However, the compression gain of Gzip and Compress when applied to V2X messages like CAMs, CPMs and MCMs has not been studied and is unclear as it will depend on the V2X message size and content (e.g. the presence of repeated substrings).

III. V2X MESSAGES

The size and statistical properties of the data to be compressed can have a high impact on the performance of the compression process. For this reason, it is important that studies are based on real data and messages. This study uses standardized V2X messages defined by ETSI for basic cooperative awareness (CAMs, Cooperative Awareness Messages), cooperative or collective perception (CPMs, Collective Perception Messages) and cooperative maneuvering or driving (MCMs, Maneuver Coordination Messages). The V2X messages have been implemented in standard-compliant testbeds developed during the H2020 TransAID project [24] and data has been obtained from field trials. The authors would like to note that the statistical properties of these messages in large deployments can vary from those obtained in these testbeds. However, all the messages analyzed in this study have been generated following the ETSI standards and hence provide useful insights into the potential of compression for V2X communications.

CAM messages are basic broadcast messages that are used to transmit information about the transmitting vehicle [25]. They contain basic information (e.g. position, speed and status information) and have a small size. CAM messages are regularly broadcasted so that vehicles improve their awareness of the driving environment. CAM messages are the basis of many services (e.g. intersection collision warning or slow vehicle indication). Automated vehicles will also broadcast CPM messages for cooperative or collective perception in connected automated driving [26]. The idea is that connected automated vehicles share information about objects detected by their on-board sensors. Using this shared information, vehicles can extend their awareness even beyond their sensors' field of view. MCM messages are being designed to implement cooperative maneuvering or driving [27]. The goal is that vehicles share their planned and desired trajectories to coordinate safely and efficiently their driving maneuvers. The planned trajectories are used by vehicles to improve the prediction of future locations of nearby vehicles and to detect conflicts, and are always included in the transmitted MCMs. The desired trajectories are used to request a coordination between vehicles. CPM and MCM messages are of larger size than CAMs and will also be continuously broadcasted. They could hence consume a high proportion of the channel bandwidth. This section summarizes the structure and format of the different messages defined (or under definition) by ETSI.

A. CAM

A CAM is composed of one ITS PDU header and multiple containers [25] (see Figure 2). The ITS PDU header is common to multiple messages. It includes data elements like the type of message and the ID of the transmitting vehicle or RSU (Road Side Unit). CAMs transmitted by vehicles must also include one Basic Container and one High Frequency Container. The Basic Container includes basic information of the vehicle that transmits the CAM, such as the vehicle type or its latitude and longitude. The High Frequency Container contains dynamic information of the transmitting vehicle, such as its heading or speed. The CAM can also contain optional containers, such as the Low Frequency Container and one or more Special Containers. The Low Frequency Container includes data elements such as the vehicle role, the path history, or the status of the exterior lights. The Special Vehicle container includes information related to the vehicle role, and was designed for public transport vehicles or emergency vehicles, among others. All containers have optional and mandatory data elements. Therefore, the CAM size depends on the number of optional containers and optional data elements considered. The CAM size can change significantly depending on the driving conditions [28].

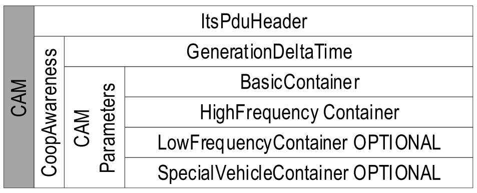


FIGURE 2. CAM structure defined by ETSI [25].

B. CPM

The CPM includes an ITS PDU header and 5 types of containers [26] (see Figure 3): a Management Container, a Station Data Container, a Sensor Information Container, a Perceived Object Containers and a Free Space Addendum Container. In addition, it also contains a data element that specifies the current number of detected objects. This number does not necessarily match with the number of objects included in the CPM because all detected objects are not included in all CPMs.

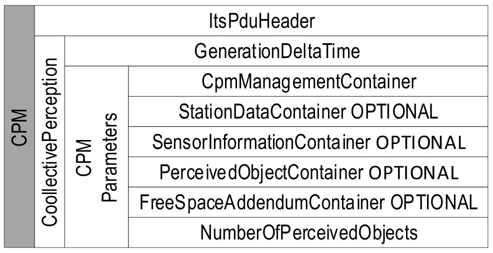

CPM
CoollectivePerception
CPM Parameters
ItsPduHeader
GenerationDeltaTime
CpmManagementContainer
StationDataContainer OPTIONAL
SensorInformationContainer OPTIONAL
PerceivedObjectContainer OPTIONAL
FreeSpaceAddendumContainer OPTIONAL
NumberOfPerceivedObjects

FIGURE 3. CPM structure defined by ETSI [26].

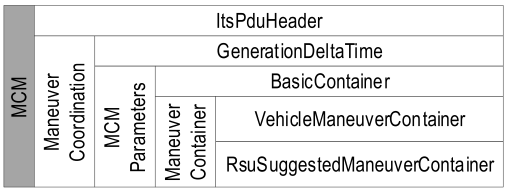


FIGURE 4. CPM structure defined by TransAID [24].

The Management Container and the Station Data Container include information about the transmitter. The Management Container is mandatory and includes its position and type (e.g. vehicle or RSU). This container also includes an optional data element to notify if the transmitted CPM is part of a larger CPM that has been segmented due to message size constraints. The Station Data Container is optional and includes additional information about the transmitting vehicle or RSU. For vehicles, this container includes information about the vehicle dynamics, such as heading, speed, angle, and its size. For RSUs, it includes information such as the Intersection Reference ID or Road Segment ID.

The Sensor Information Container, the Perceived Object Container and the Free Space Addendum Container are used to exchange information about the on-board sensors and the perceived environment (detected objects and free space). The Sensor Information Container is optional and informs about the on-board sensors used by the transmitter (up to 128 sensors). For each sensor, this container includes the sensor ID, sensor type (e.g. camera, lidar or radar) and its detection area. The receiver can use this information to estimate the areas covered by the sensors of the transmitter. The Perceived Object Containers is also optional and includes information about the detected objects (up to 128 detected objects). For each object, it includes information such as the object ID, position, speed, acceleration, and size of the object, among other fields. Finally, the Free Space Addendum Container is optional and describes the free space areas within the sensor detection areas. The receiver can use this information to better estimate the free space areas around the transmitting vehicle.

C. MCM

The standardization of the MCM has not yet concluded [27] but the H2020 TransAID project has proposed a format for MCM messages [24], [29] that follows current discussions at ETSI. We use this format in this study since it is the only concrete proposal to date. The MCM includes the ITS PDU header and the containers illustrated in Figure 4. The Basic Container includes the geographic position and type of the transmitter (a vehicle or an RSU). The Maneuver Container can include a Vehicle Maneuver Container if it is transmitted by a vehicle or an RSU Suggested Maneuver Container if it is transmitted by the road infrastructure.

Vehicles transmit the Vehicle Maneuver Container, which must always include the planned trajectory of the vehicle. In addition, this container can include its desired trajectory. Each trajectory contains a variable number of trajectory points, each of them with its coordinates relative to the vehicle position (*deltaXCm* and *deltaYCm*), the remaining time to reach the point (*deltaTimeMs*), and the vehicle heading and speed when it reaches the point (*headingValue* and *absSpeed*). The container also includes information such as the heading, speed or acceleration.

The RSU Suggested Maneuver Container is used by RSUs to support maneuver coordination. This container includes a list of driving advice or suggestions sent to vehicles to help them coordinate the maneuvers (i.e. the current proposal envisions the road infrastructure to support and not control the driving). Four types of advice have been defined: car following advice, lane advice, Transition of Control advice, and safe spot advice [24].

IV. EXPERIMENTAL TESTBEDS FOR THE COLLECTION OF V2X MESSAGES

The V2X messages used in this study have been obtained from field trials carried out using standard-compliant testbeds developed during the H2020 TransAID project [24]. In TransAID, different CAV and RSU prototypes have been implemented (Figure 5) that have been used to demonstrate how infrastructure-assisted traffic management solutions can reduce safety risks and traffic disruptions. The developed CAV and RSU prototypes are capable of exchanging basic awareness information (CAMs), information about detected objects (CPMs) and information to manage and coordinate maneuvers (MCMs) using V2X communications. The CAVs and RSUs are equipped with a V2X module that enables V2V and V2I communications. The V2X module is implemented using a Cohda Wireless's MK5 (see Figure 5b). The MK5 is compliant with the latest ETSI standards for V2X communications at the different layers of the ETSI ITS Architecture [11], including the ITS-G5 radio access technology.

CAMs and MCMs were collected in field trials that showcased the maneuver coordination using V2V when CAVs performs a lane merge in a highway entrance scenario. In this scenario, one CAV was driving on the highway while another CAV entered through an on ramp with certain risk of collision. These two CAVs periodically exchanged CAMs and MCMs to detect each other and coordinate their maneuvers to perform the lane merge maneuver efficiently

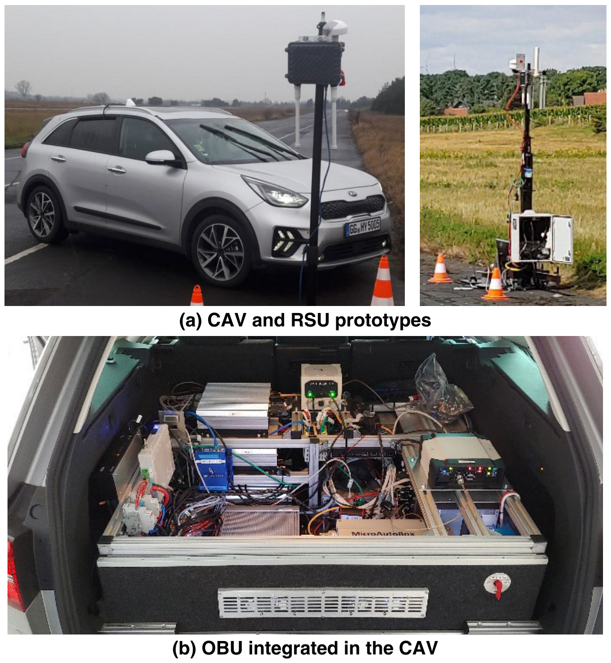


FIGURE 5. Prototypes used to collect V2X messages.

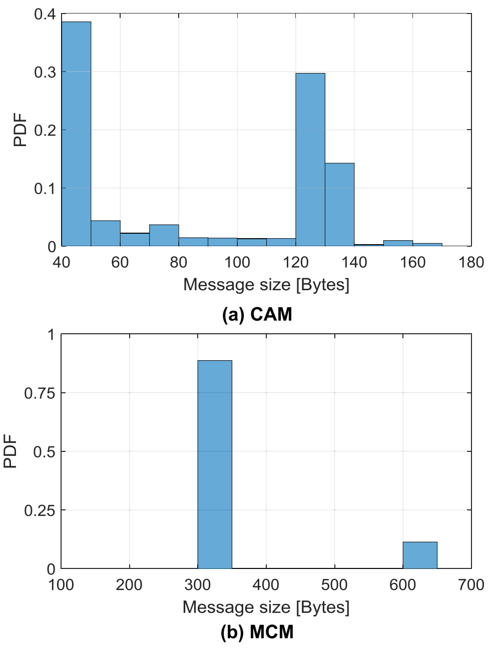


FIGURE 6. PDF (Probability Density Function) of the size of CAMs and MCMs.

and safely. The CAVs combine the information collected from the V2X messages with the information perceived by their on-board sensors. The combined information is used by the Autonomous Driving Software (AD SW) module to plan and execute the CAV's autonomous maneuvers. In addition, the AD SW module provides information to the V2X module to create the V2X messages to be transmitted (e.g. the planned and desired trajectories).

CPMs were generated by an RSU installed at the Tostmannplatz intersection in Braunschweig. This intersection has four approaches which are controlled by traffic lights. There are two lanes per direction on the main road and additional lanes for left-turning. The RSU uses an hemispheric camera of type Samsung PNM-9020V [30] to periodically generate CPMs that contain information about the detected objects. This information was also used as input to a Traffic Monitoring module to derive the infrastructure-assisted traffic management measures based on context conditions.

V. COMPRESSION OF V2X MESSAGES

This section analyzes the compression gain that can be achieved for the three V2X message types (CAMs, CPMs and MCMs). To this aim, we first analyze the statistical properties of the V2X messages collected in the field trials using the experimental platforms described in the previous section. We then compare the compression gain that we achieve with the three compression algorithms. We identify the best compression algorithm for the V2X messages, and we compare the compression gains achieved by this algorithm to its theoretical compression limit. Finally, this section quantifies the time needed to compress and decompress the considered V2X messages.

A. STATISTICAL PROPERTIES OF V2X MESSAGES

Figure 6 depicts the PDF (Probability Density Function) of the size of the CAMs collected during the field experiments. Most of the collected CAMs have a message size of around 40 Bytes or 130 Bytes. The CPMs used in this study were generated and transmitted by an RSU located in an intersection. All CPMs collected have a size of 105 Bytes. The collected MCMs were transmitted by 2 vehicles in a merging highway scenario. All MCMs included the planned trajectory. When a maneuver coordination was required, they included both the planned and the desired trajectories. As a consequence, the collected MCMs have 2 different sizes (329 Bytes or 608 Bytes) depending on whether they contain the desired trajectory or not. Figure 6b shows that it was more common for vehicles to broadcast MCMs without a desired trajectory during the trials.

We have analyzed each V2X message type considering three different alphabets. The alphabets contain $K = 2^4 = 16$, $K = 2^8 = 256$ and $K = 2^{12} = 4096$ symbols, respectively:

- $K = 16$. Each symbol is represented with 4 bits without compression. The alphabet can also be represented as the set of hexadecimal symbols (0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E and F) or as integers (from 0 to 15).
- $K = 256$. Each symbol is represented with 8 bits without compression. The alphabet can also be represented as the set of all pairs of possible hexadecimal symbols (00, 01, 02, ..., FE and FF) or as integers (from 0 to 255).
- $K = 4096$. Each symbol is represented with 12 bits without compression. The alphabet can also be represented the set of all groups of 3 possible hexadecimal symbols (000, 001, 002, ..., FFE and FFF) or as integers (from 0 to 4095).

We have analyzed all collected messages to calculate the probability of each symbol per V2X message type and alphabet. Figures 7, 8 and 9 show the PDF of the symbols for the collected CAMs, CPMs and MCMs. The case of $K = 4096$ is not shown due to visibility reasons as it has too many symbols. Figures 7, 8 and 9 show that some symbols of the alphabet have higher probability than others, which increases the potential to achieve large compression gains.

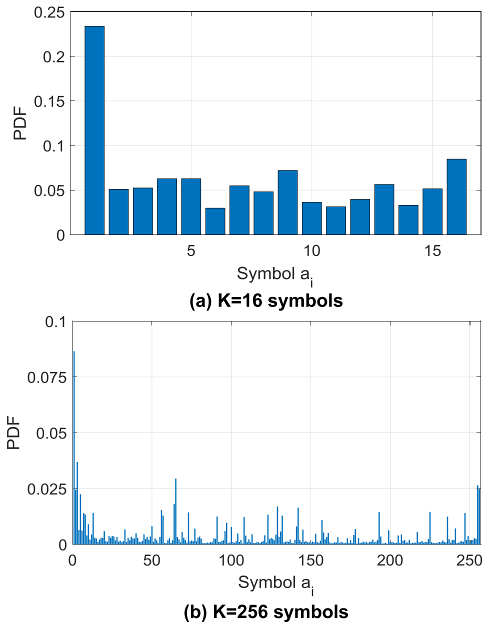


FIGURE 7. **PDF (Probability Density Function) of the symbols in CAMs.**

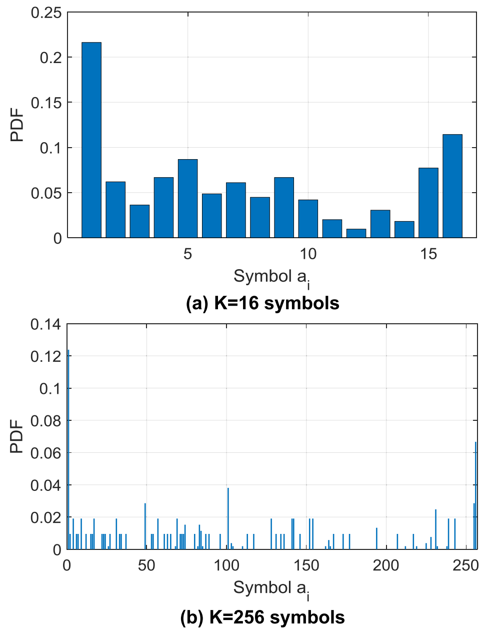


FIGURE 8. **PDF (Probability Density Function) of the symbols in CPMs.**

B. COMPRESSION GAINS

This section compares the compression gain that can be achieved for CAMs, CPMs and MCMs messages using the Gzip, Compress and the Shannon-Fano algorithms. Gzip and Compress are based on adaptive dictionary compression while Shannon-Fano is based on entropy compression. For Shannon-Fano, we consider implementations with different alphabet sizes ($K = 16$, 256 and 4096). We refer to these implementations as sf16, sf256 and sf4096.

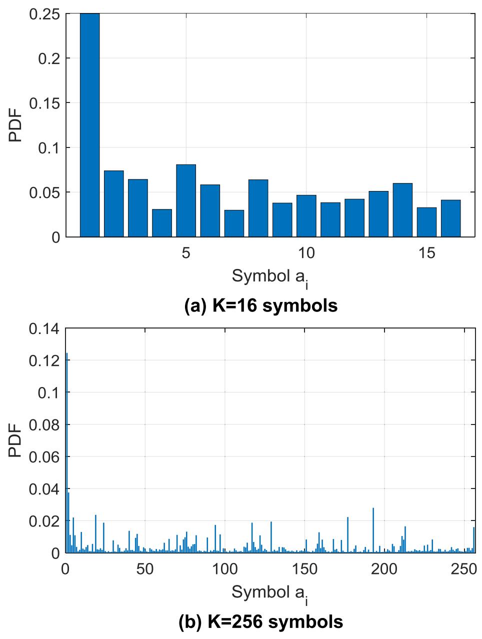


FIGURE 9. **PDF (Probability Density Function) of the symbols in MCMs.**

Gzip and Compress do not require any pre-processing and can be applied to each message directly. On the other hand, Shannon-Fano compression requires building the dictionaries of codewords beforehand. We have used existing open source libraries for Gzip (version 1.5) and Compress (version 4.2.4.4) under CentOS Linux release 7.4.1708. We have implemented our own source code to construct the dictionaries of codewords in Shannon-Fano. We then use existing search and replace libraries to replace message symbols with their corresponding codewords. To this aim, we use as input the PDF of the symbols (Figures 7, 8 and 9). Shannon-Fano algorithm assigns a higher number of bits to those symbols with lower probability. As an example, Table 1 presents the dictionaries of codewords for CAMs, CPMs and MCMs when applying the Shannon-Fano algorithm for $K = 16$ symbols. Without compression, 4 bits are necessary to represent each symbol. This number of bits is reduced with compression for the symbols with higher probability (e.g. symbol "0"). The same procedure has been followed for $K = 256$ and $K = 4096$, but their dictionaries are not shown due to their size.

Figure 10 shows the compression gain achieved with the different algorithms for CAMs, CPMs and MCMs. The bars represent the average values and the vertical lines the 5th and 95th percentiles. Positive compression gains mean that it was possible to compress messages while negative ones indicate that the message size actually increased after compression.

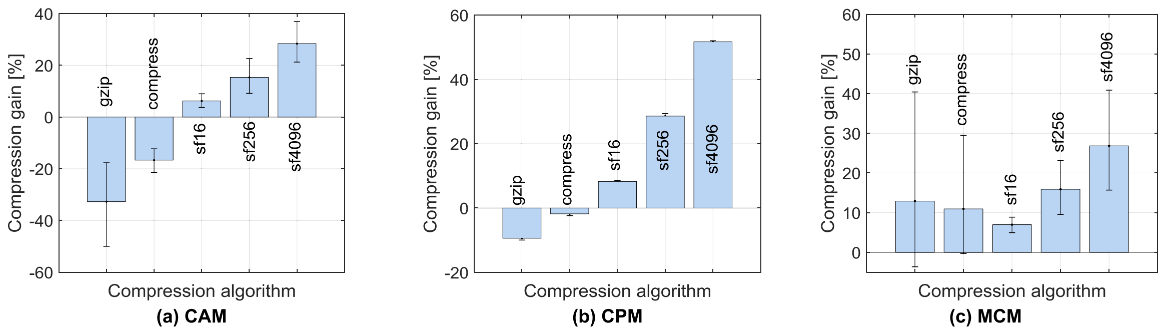


FIGURE 10. Compression gain.

TABLE 1. Dictionary of Codewords for CAM, CPM and MCM using Shannon-Fano algorithm with K=16.

Symbol	CAM	CPM	MCM
0	00	00	00
1	010	011	0110
2	1000	100	0111
3	1010	1010	11110
4	11000	10111	010
5	11011	11001	1010
6	11001	11010	11111
7	1011	1110	1000
8	011	010	1110
9	111110	11000	10111
A	1001	10110	11011
B	11111	111111	1100
C	11010	111110	10110
D	11110	11110	1001
E	11100	11101	11101
F	11101	11011	11010

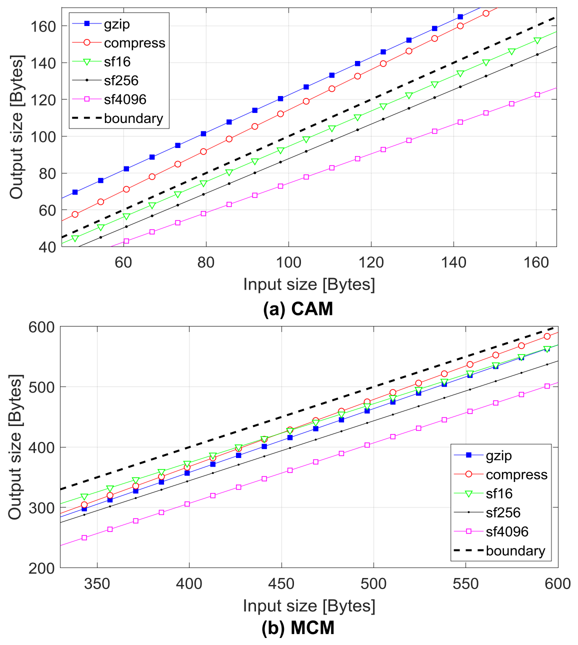


FIGURE 11. Size of compressed message (output) as a function of the size of the uncompressed message (input).

Figure 10 clearly shows that the Shannon-Fano algorithm is always capable to compress all V2X messages with the compression gain increasing with the size of the alphabet K. With the largest alphabet, Shannon-Fano always compresses on average all messages by more than 25%. On the other hand, Gzip and Compress (adaptive dictionary compression) can only compress MCMs. This is the case because CAMs and CPMs have multiple containers without necessarily repeated patterns. Gzip and Compress can compress MCMs (Figure 10c) because planned and desired trajectories occupy a large part of the MCM. Each trajectory can have 30 trajectory points, and thus MCMs can have long sequences of repeated or similar values when e.g. the vehicle is moving at constant speed and heading.

The message size can impact the compression gain. Figure 11 depicts the output size (S_o) –after compression– as a function of the input size (S_i) –before compression– for CAMs and MCMs.[1] The dashed line represents the case where $S_o = S_i$. All values below the dashed lines represent a successful compression (i.e. positive compression gain). All values above the dashed lines represent scenarios where the compression algorithm increased rather than decreased the message size. The solid lines represent the linear models that best fit the output-input values for each compression algorithm and message. These models are expressed as:

$$S_o = \alpha \cdot S_i + \beta \quad (1)$$

[1]All CPMs had the same size in our experiments. This is why Figure 11 does not represent the case for CPMs. The size of CPMs depends on the number of sensors and the number of connected automated vehicles in the scenario. Since this number was small in our testbed, there is no variability for CPMs. However, future CPMs in scenarios with many more automated vehicles will be more variable and hence higher compression gains than those observed in our experiments could be achieved.

TABLE 2. Parameters of the V2X message compression models.

Compression algorithm	CAM		CPM		MCM	
	α	β	α	β	α	β
Gzip	1.022	20.32	1.094	0	1.056	-64.4
Compress	1.098	4.572	1.018	0	1.111	-76.66
sf16	0.961	-1.536	0.917	0	0.975	-15.5
sf256	0.940	-6.158	0.714	0	0.992	-52.44
sf4096	0.799	-5.472	0.483	0	1.001	-93.41

where α and β are the parameters of the models and their values are presented in Table 2. Parameter β is equal to zero for CPMs because the same compression was achieved for all collected CPM messages[1]. The community can use the linear models plotted in Figure 11 to consider the impact of compression as a function of the original message without having to implement the compression algorithms. We use Figure 11 to derive Figure 12 that depicts the compression gain as a function of the size of the input messages. The figure shows that the Shannon-Fano algorithm always achieves a positive compression gain, but the gain slightly decreases as the size of the uncompressed message (input) increases. This trend is observed for both CAMs and MCMs, and, is due to the fact that most of the messages have a small size (see Figure 6). As a consequence, the constructed codewords are better suited to the symbol probabilities of the small messages, and therefore provide better compression for these messages. Figure 12 shows that Gzip and Compress experience the same trend for MCM messages and also achieve a positive compression gain. In general, the Shannon-Fano algorithm outperforms Gzip and Compress except when the size K of the alphabet is very low. Gzip and Compress achieve opposite trends with CAMs and their compression gain improves with the input size. This is the case because the probability to find repeated substrings is higher when the input size is larger. However, the compression gain remains always negative and the Shannon-Fano algorithm clearly outperforms Gzip and Compress. This is the case because Gzip and Compress add headers that reduce the compression gain, and also because they are designed to achieve good compression gains when messages are of larger size than CAMs. This is already partly visible in Figure 12b when considering MCMs that are larger than CAMs.

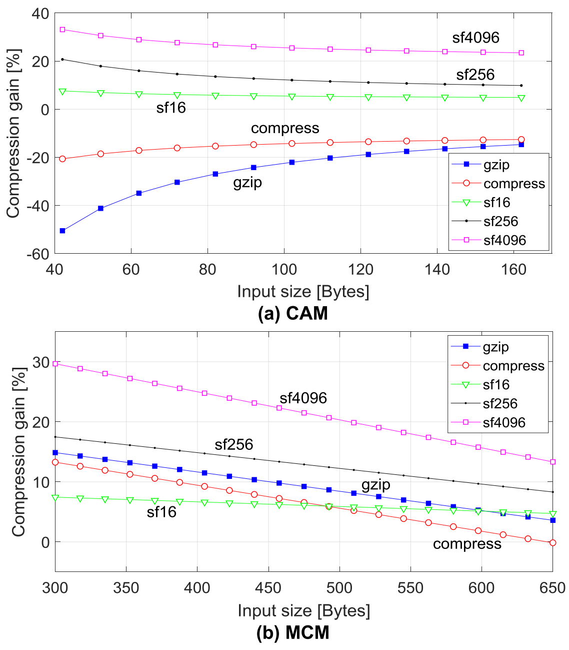


FIGURE 12. Compression gain as a function of the message size.

C. COMPRESSION LIMIT

The previous section has demonstrated that the Shannon-Fano algorithm achieves the highest compression gain. We then derive for this algorithm its theoretical compression limit to check how close our implementation is to this limit.

The compression limit for entropy compression is established by the Shannon's source coding theorem. This theorem shows that it is impossible to compress the data such that the average number of bits per symbol is less than the Shannon entropy of the source that generates the symbols [17]. In other words, n independent and identically-distributed random variables each with entropy $H(X)$ can be compressed into more than $n{\cdot}H(X)$ bits with negligible risk of information loss [17]. This limit can be useful in this study to estimate how close the tested algorithms are to this limit.

To calculate the data compression limit, we must first define the Shannon entropy of a source $H(X)$. The entropy of a memoryless source can be defined as the average amount of information acquired by observation of a single symbol on the source output [18]. The amount of information acquired by observation of a given symbol a_i is given by the formula:

$$I(a_i) = \log_2\left(\frac{1}{P(a_i)}\right) \tag{2}$$

where $P(a_i)$ is the probability that the source generates the symbol a_i. In our study, this probability is equivalent to the probability that the symbol a_i appears in a V2X message. We consider that the source has an alphabet of K possible symbols, $X = \{a_1, \ldots, a_K\}$, and each of them is generated with probability $P(a_1), \ldots P(a_K)$. Therefore, the Shannon entropy of a given source can be calculated as:

$$H(X) = E\left[I\left(a_i\right)\right] = \sum_{i=1}^{K} P\left(a_i\right) \log_2\left(\frac{1}{P\left(a_i\right)}\right) \tag{3}$$

Without compression, each symbol is represented by $\log_2(K)$ bits, and therefore the number of bits needed to represent a V2X message with n symbols is:

$$B_{nc} = n \cdot \log_2(K) \tag{4}$$

For example, if the alphabet contains $K = 16$ symbols, each symbol can be represented by $log_2(16)$=4 bits, from 0000 to 1111. As a result, a V2X message with $n = 200$ symbols can be represented by $B_{nc} = 800$ bits without compression. Given Shannon's source coding theorem, the minimum number of bits of the message is bounded by:

$$B_c = n \cdot H(X) \tag{5}$$

The compression gain limit of a V2X message with n symbols that can be then expressed as a percentage:

$$CG_{\text{lim}}[\%] = 100 \cdot \frac{B_{nc} - B_c}{B_{nc}} = 100\left(1 - \frac{H(X)}{\log_2(K)}\right) \tag{6}$$

This limit does not depend on the length of the V2X message. It depends on the number of symbols in the alphabet (K) and the entropy ($H(X)$), which in turn depends on the probabilities of the different symbols.

To derive the compression limits, we compute the entropy $H(X)$ for V2X each message type and value of K using equation (3) and the PDF of the symbols (Figures 7, 8 and 9). Then, we compute with equation (6) the theoretical limit of the compression gain achievable with entropy compression, i.e. with the Shannon-Fano algorithm. Table 3 reports this limit per V2X message type and size K of the alphabet. The table also reports the entropy values computed. The compression gain increases with the number of symbols in the alphabet. For $K = 4096$, compression gain limits up to approximately 50% could be achieved, which highlights again the potential of entropy compression to reduce the channel load and improve the V2X communications reliability. Larger alphabets would be possible and would provide higher compression gains at the expense of processing power. It is important to highlight that the Shannon-Fano algorithm implemented for this study obtains average compression gains (Figure 10) that perfectly match with the theoretical compression limits shown in Table 3. This further validates our implementation and highlights again the potential of V2X message compression.

TABLE 3. Entropy and theoretical limit of the compression gain of entropy compression.

K	Entropy $H(X)$			Theoretical compression gain		
	CAM	CPM	MCM	CAM	CPM	MCM
16	3.73	3.65	3.69	6.7%	8.6%	7.7%
256	6.89	5.66	6.73	13.8%	29.2%	15.9%
4096	8.68	5.73	8.85	27.6%	52.2%	26.2%

D. COMPRESSION AND DECOMPRESSION TIME

The previous sections have shown that compression can provide significant gains to V2X communications by decreasing the size of the transmitted messages and consequently reducing the channel load. However, compression and decompression requires some processing time that must be small given the low latency requirements of vehicular communications. This section evaluates the time needed to compress and decompress each V2X message type with the different algorithms considered. For this evaluation, we have used an off-the-shelf Intel Xeon Gold 6130 CPU with 2.10GHz base frequency, 22MB of L3 cache size and 2666 MHz of maximum memory speed. We have used existing open source libraries for Gzip and Compress, but have implemented our own source code to evaluate the Shannon-Fano algorithm.

Figure 13 plots the compression and decompression times obtained with the different algorithms. The times are depicted separately for CAMs, CPMs and MCMs. The bars in the figure represent the average values and the vertical lines represent the 5th and 95th percentiles. Figure 13 shows that Gzip and Compress achieve the lowest compression and decompression times (between 3 ms and 8 ms approximately) independently of the message type. The Shannon-Fano algorithm can achieve similar compression and decompression times to those observed with Gzip and Compress when $K = 16$ symbols. However, the time needed to compress and decompress with the Shannon-Fano algorithm significantly increases when the number of symbols K increases because of the larger number of codewords. The obtained results in Figure 10 and Figure 13 clearly show the existing trade-off between computing time and compression gain, since larger compression gains are achieved with higher values of K.

It is also interesting to highlight in Figure 13 that each algorithm achieves compression and decompression times that are of the same order of magnitude, independently of the message type. However, the number of messages that a vehicle needs to compress at the transmitter side is

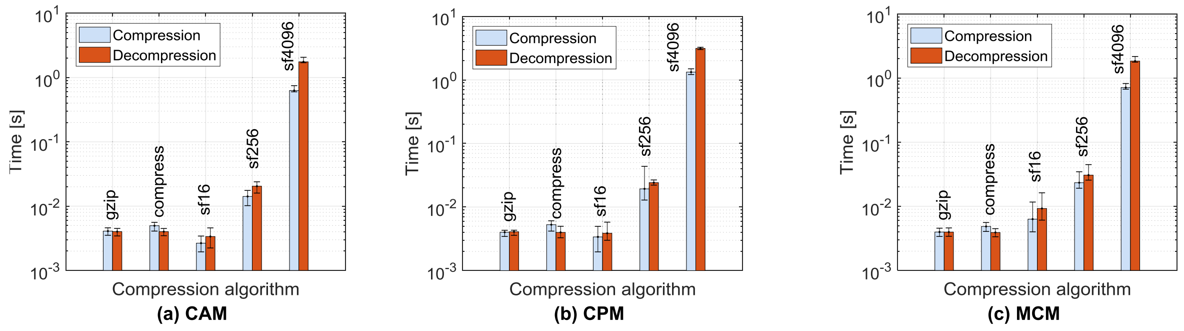


FIGURE 13. Compression and decompression time.

significantly lower than the number of messages that it receives from all neighboring vehicles and has to decompress at the receiver side. At the transmitter side, a delay of a few milliseconds (e.g. below 10 ms) created by message compression could be tolerated given that the time between V2X message transmissions is typically equal or higher than 100 ms [31]. However, at the receiver side, a decompression time below 1 ms could be needed to be able to decompress all V2X messages in real time under highly loaded scenarios. This is the case because the number of V2X messages that can be successfully received per second using IEEE 802.11p in a 10-MHz channel is around 1200 for a channel load corresponding to a Channel Busy Ratio (CBR) of 60%[2] [32]. The CBR is defined as the percentage of time that the radio channel is sensed as busy. To decompress 1200 messages per second in real time, the average decompression time of a message should be lower or equal than $1/1200 = 0.83$ ms. The results presented in this study show the potential of V2X message compression but also reveal that further work is needed for its real time implementation, e.g. through dedicated hardware and software implementations that reduce the decompression time.

VI. IMPACT OF MESSAGE COMPRESSION ON V2X COMMUNICATIONS

This section analyses the impact of V2X message compression on the reliability of V2X communications. To this aim, we have conducted a simulation study using the network simulator ns-3. All vehicles are equipped with an ITS-G5 transceiver based on IEEE 802.11p and operate in the same channel. All vehicles generate and transmit CAMs, CPMs and MCMs. CAMs are generated following the ETSI generation rules [25]. These rules specify that CAMs should be generated every 100ms to 1s. A vehicle should generate a new CAM if any of the following triggering conditions is satisfied [25]:

- The distance between its current position and the position included in its previous CAM exceeds 4 m.
- The absolute difference between its current speed and the speed included in its previous CAM exceeds 0.5 m/s.
- The absolute difference between its current heading and the heading included in its previous CAM exceeds 4°.
- The time elapsed since the last CAM was generated is equal to or higher than 1 s.

The size of each CAM is randomly selected following the PDF of Figure 6.

CPMs are also generated following the ETSI generation rules [26] that define how often a vehicle should generate a CPM and what information should be included in each CPM. A vehicle should check every *T_GenCpm* if a new CPM should be generated, where 100 ms $\leq$ *T_GenCpm* $\leq$ 1000 ms. A vehicle should generate a new CPM if it has detected a new vehicle, or if any previously detected vehicle satisfies any of the following conditions:

- its absolute position has changed by more than 4 m since the last time its data was included in a CPM;
- its absolute speed has changed by more than 0.5 m/s since the last time its data was included in a CPM;
- its absolute velocity has changed by more than 4° since the last time its data was included in a CPM;
- the last time it was included in a CPM was 1 (or more) seconds ago.

In a new CPM, the vehicle includes all new detected vehicles and those previously detected vehicles that satisfy at least one of the previous conditions. The on-board sensor used in this study for the detection of vehicles is a 360° sensor with 150 m range [26]. In the simulation, the size of each CPM depends on the number of detected objects and the generation rules. The MCM generation rules are still under definition. We have then considered that all vehicles transmit MCMs at a fixed rate equal to 5 Hz. The size of each MCM is randomly selected following the PDF of Figure 6.

When V2X message compression is enabled, the size of the compressed message is calculated as a function of the original uncompressed message using the models presented in Figure 11 and Table 2.

The traffic scenario is a six-lane highway (three lanes per each direction) with 5 km length and a lane width of 4 meters. To avoid boundary effects, statistics are only taken from the vehicles located in the 2 km around the center of the simulation scenario. We simulate three different traffic densities: 60 veh/km, 120 veh/km and 180 veh/km. The configuration of the scenario is summarized in Table 4. The propagation effects are modelled using the Winner+ B1 propagation model following 3GPP guidelines in [33]. The communication parameters are summarized in Table 5.

TABLE 4. Scenario parameters.

Parameter	Traffic density		
	Low	Medium	High
Traffic density	60 veh/km	120 veh/km	180 veh/km
Speed per lane	140 km/h 132 km/h 118 km/h	70 km/h 66 km/h 59 km/h	50 km/h
Number of lanes	6 (3 per driving direction)		
Highway length	5 km		

The impact of V2X message compression on V2X networks is first analyzed by means of the channel load. The channel load is measured using the CBR. Table 6 shows the average CBR obtained for the three traffic densities considered and all compression algorithms evaluated. The table shows between parentheses the relative CBR difference with the scenario without compression. Table 6 shows that the highest reduction of the channel load at the network level is achieved with sf4096. This is in line with the results presented

[2]We consider a 60% target since congestion control protocols, like DCC in ITS-G5, control and maintain the load around this target value.

TABLE 5. Communication parameters.

Parameter	Value
Transmission power	23 dBm
Antenna gain (tx and rx)	0 dBi
Channel bandwidth/carrier freq.	10 MHz / 5.9 GHz
Noise figure	9 dB
Energy detection threshold	-85 dBm

TABLE 6. Average CBR (Chanel Busy Ratio) experienced.

Compression	Traffic density		
	Low	Medium	High
None	41.80%	61.80%	73.31%
Gzip	42.54% (+1.8%)	62.08% (+0.5%)	73.43% (+0.2%)
Compress	41.45% (-0.8%)	60.98% (-1.3%)	72.46% (-1.2%)
sf16	39.80% (-4.8%)	59.51% (-3.7%)	71.18% (-2.9%)
sf256	35.70% (-14.6%)	54.64% (-11.6%)	66.61% (-9.1%)
sf4096	30.61% (-26.8%)	48.31% (-21.8%)	60.36% (-17.7%)

in the previous section. sf4096 can reduce the CBR up to around 27% in the best scenario. This is a non-negligible result given that data compression does not reduce the amount of information transmitted or the number of messages transmitted. It is interesting to note that the average reduction of the CBR is not equal to the average compression gain achieved. For example, the average compression gain with sf4096 was between 27% and 52% approximately (Figure 10) but the CBR is reduced between 18% and 27%. This effect is produced because the compression is applied at the Facilities layer, and the headers added at the Transport & Network, Access and PHY layers are not compressed, to minimize the impact on existing standards and facilitate its practical implementation. It is also worth noting that the relative reduction of the CBR decreases if the traffic density increases. This effect is related to packet collisions. When the traffic density and the CBR increase, the number of packet collisions also augment. When two or more packets overlap in time, their contribution to the CBR is reduced. When no compression is applied, higher packet collisions are produced, especially for the high traffic density scenario. This explains why the CBR improvement decreases with the traffic density, although it is clear that in this case it is not a positive effect.

The reduction of the channel load has a positive effect on the reliability of V2X communications. When the channel load decreases thanks to message compression, the number of packets that are lost due to packet collisions and interferences also decreases. The improvement in reliability is observed when analyzing the PDR (Packet Delivery Ratio) that is defined as the probability of correctly receiving a V2X message at a certain distance to the transmitter. Figure 14 plots the PDR experienced in the low, medium and high traffic density scenarios when using the different compression algorithms.

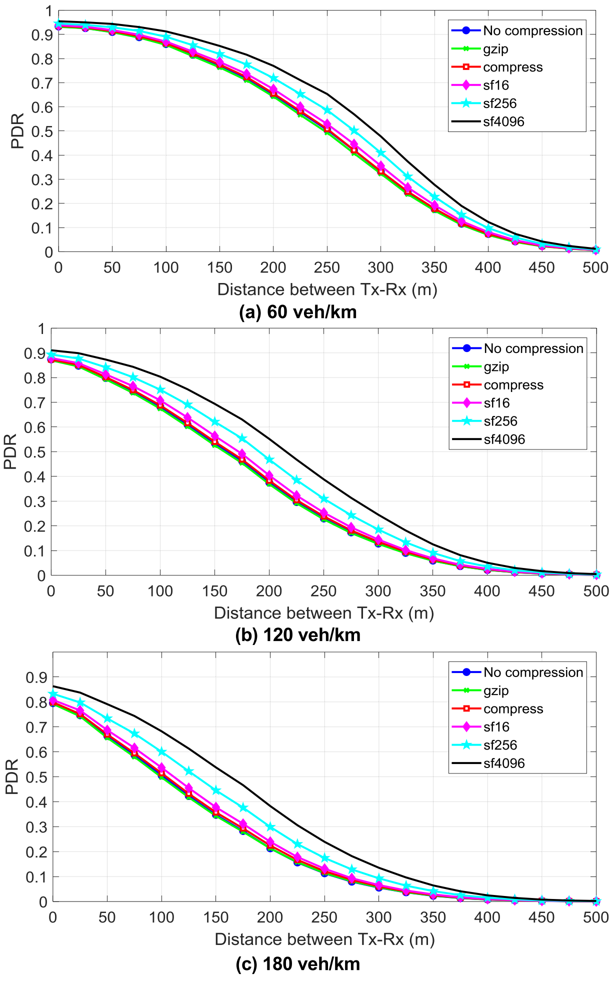


FIGURE 14. PDR (Packet Delivery Ratio) with and without V2X message compression.

The PDR is shown as a function of the distance between transmitter and receiver. The compression algorithms with highest compression gains (and thus higher CBR reductions) achieve the highest improvement of PDR compared to the scenario without compression. This improvement augments the distance at which a connected vehicle can be detected with CAMs, the distance at which an object can be detected with CPMs, or the distance at which a maneuver coordination can safely take place with MCMs.

A common metric to compare the reliability of different solutions is the distance at which a given PDR threshold is achieved. This distance is shown in Figure 15 for a PDR threshold of 0.7 to more clearly show the gain achieved with data compression. This figure shows, for example, that in the high-density scenario the distance at which a PDR of 0.7 is obtained is more than double with sf4096 compared with the scenario without compression. Figures 16 and 17 plot the communications range improvement that would be achieved with different PDR thresholds thanks to V2X message compression. The communications range improvement is computed in Figure 16 as the difference of the distances

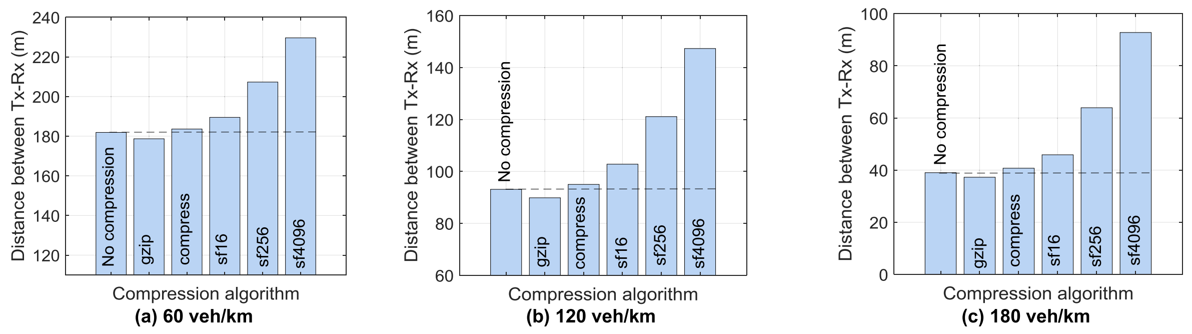


FIGURE 15. Distance between Tx-Rx at which a PDR threshold of 0.7 is achieved.

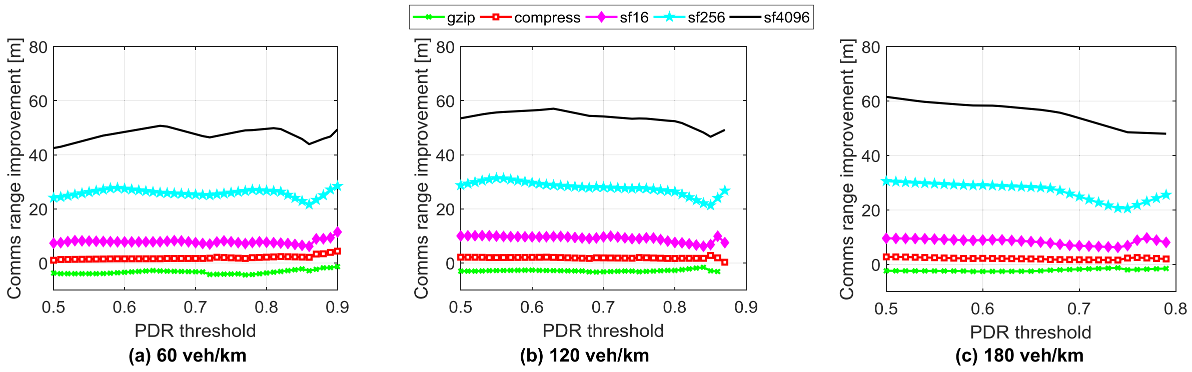


FIGURE 16. Communications range improvement (in meters) achieved with V2X message compression for different PDR thresholds.

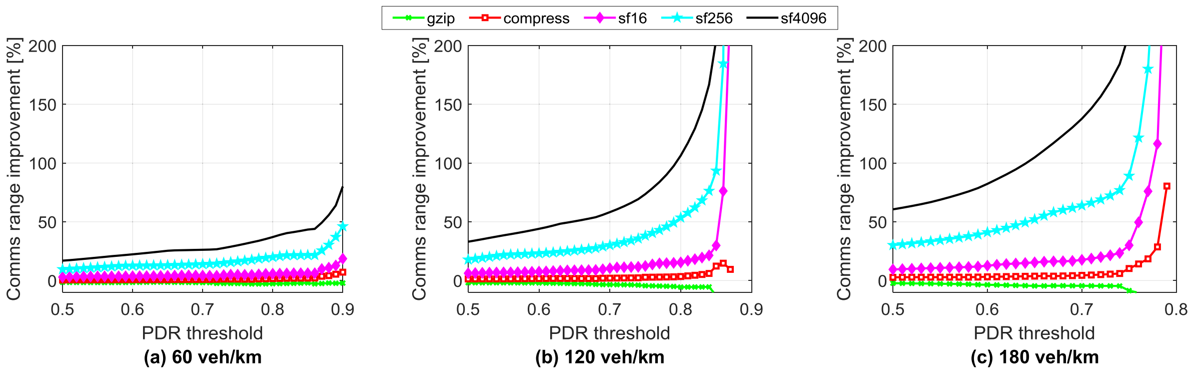


FIGURE 17. Relative communications range improvement (in percentage) achieved with V2X message compression for different PDR thresholds.

at which the corresponding PDR threshold is achieved with and without compression. It is therefore expressed in meters. The results obtained show that the PDR threshold does not significantly influence the communications range improvement when measured in absolute values. Figure 17 plots the relative communications range improvement as a percentage. As it can be observed, the improvement increases as the PDR threshold increases. This is the case because the communications range decreases as the PDR threshold increases. The obtained results demonstrate that the use of V2X message compression significantly increases the distance at which a given PDR is achieved. It should be reminded that this is achieved without modifying the communication parameters (hence, without reducing the communications range) or the message rate.

VII. CONCLUSION

This paper proposes the compression of V2X messages to reduce the communication channel load and improve the scalability and reliability of vehicular networks. Contrary to conventional congestion control protocols, V2X message compression can reduce the channel load without modifying the communication parameters (and hence the range) or the message rate. This has a positive effect on the execution of applications compared to current congestion control protocols that are based on packet dropping [7]. The obtained results have shown that V2X messages (CAM, CPM and MCM) can be compressed up to around 40-50%. This compression is performed at the upper layers and is therefore agnostic from the underlaying radio access technology. We have also demonstrated that this compression can reduce the channel load and hence improve the packet delivery ratio and thus the communications range. The gains achieved by compression depend on the compression algorithm utilized and the V2X message type and size. The selected algorithm also impacts the compression and decompression times and a balance between compression gains and compression times is necessary for supporting low latency V2X communications. The compression algorithms are independent of the underlying radio access technologies. However, 3GPP-based technologies such as LTE-V2X or 5G NR V2X organize differently the radio resources and the access to the wireless medium. It would then be interesting that future studies quantify the gains (channel load and reliability among others) that compression can bring to LTE-V2X and 5G NR V2X.

REFERENCES

[1] *Intelligent Transport Systems (ITS); Decentralized Congestion Control Mechanisms for Intelligent Transport Systems Operating in the 5 GHz Range; Access Layer Part*, Standard TS 102 687 v1.2.1, ETSI TC ITS, Apr. 2018.

[2] O. Amador, I. Soto, M. Calderon, and M. Uruena, ''Experimental evaluation of the ETSI DCC adaptive approach and related algorithms,'' *IEEE Access*, vol. 8, pp. 49798–49811, 2020.

[3] A. Hisham, E. G. Strom, F. Brannstrom, and L. Yan, ''Scheduling and power control for V2 V broadcast communications with co-channel and adjacent channel interference,'' *IEEE Access*, vol. 7, pp. 67041–67058, 2019.

[4] F. Goudarzi and H. Asgari, ''Non-cooperative beacon power control for VANETs,'' *IEEE Trans. Intell. Transp. Syst.*, vol. 20, no. 2, pp. 777–782, Feb. 2019.

[5] B.-M. Cho, M.-S. Jang, and K.-J. Park, ''Channel-aware congestion control in vehicular cyber-physical systems,'' *IEEE Access*, vol. 8, pp. 73193–73203, Apr. 2020.

[6] M. Sepulcre, J. Gozalvez, and B. Coll-Perales, ''Why 6 mbps is not (Always) the optimum data rate for beaconing in vehicular networks,'' *IEEE Trans. Mobile Comput.*, vol. 16, no. 12, pp. 3568–3579, Dec. 2017.

[7] M. Sepulcre, J. Mira, G. Thandavarayan, and J. Gozalvez, ''Is packet dropping a suitable congestion control mechanism for vehicular networks?'' in *Proc. IEEE 91st Veh. Technol. Conf. (VTC-Spring)*, May 2020, pp. 25–28.

[8] G. Karagiannis, O. Altintas, E. Ekici, G. Heijenk, B. Jarupan, K. Lin, and T. Weil, ''Vehicular networking: A survey and tutorial on requirements, architectures, challenges, standards and solutions,'' *IEEE Commun. Surveys Tuts.*, vol. 13, no. 4, pp. 584–616, 4th Quart., 2011.

[9] IETF. (1999). *Hypertext Transfer Protocol – HTTP/1.1*. The Internet Society. [Online]. Available: https://tools.ietf.org/html/rfc2616

[10] B. A. King, *Speed Up Your Site: Web Site Optimization*. Indianapolis, IN, USA: New Riders Press, Jan. 2003.

[11] *Intelligent Transport Systems (ITS); Communications Architecture*, Standard ETSI EN 302 665 V1.1.1, ETSI TC ITS, 2010.

[12] *Intelligent Transport Systems—Communications Access for Land Mobiles (CALM)—Architecture*, document ISO 21217, ISO/TC 204, Intelligent transport systems, 2014.

[13] *Wireless Access in Vehicular Environments (WAVE)—Architecture*, Standard IEEE 1609.0-2019, IEEE Standards Association, Apr. 2019.

[14] S. R. Kodituwakku and U. S. Amarasinghe, ''Comparison of lossless data compression algorithms for text data,'' *Indian J. Comput. Sci. Eng.*, vol. 1, no. 4, pp. 416–426, 2010.

[15] *H2020 TransAID Project. Grant Agreement No 723390*. Accessed: Apr. 2020. [Online]. Available: https://www.transaid.eu/

[16] M. Nelson and J.-L. Gailly, *The Data Compression Book*, 2nd ed. Hoboken, NJ, USA: Wiley, Dec. 1995.

[17] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge, U.K.: Cambridge Univ. Press, 2003.

[18] K. Wesołowski, *Introduction to Digital Communication Systems*. Austin, TX, USA: Whiley, 2009.

[19] *Ubuntu Documentation, Compress*. Accessed: May 2020. [Online]. Available: http://manpages.ubuntu.com/manpages/xenial/man1/compress.1.html

[20] *GNU Documentation: Gzip*. Accessed: May 2020. [Online]. Available: https://www.gnu.org/software/Gzip/

[21] J. Ziv and A. Lempel, ''A universal algorithm for sequential data compression,'' *IEEE Trans. Inf. Theory*, vol. IT-23, no. 3, pp. 337–343, May 1977.

[22] IETF. (1996). *GZIP File Format Specification Version 4.3*. The Internet Society. [Online]. Available: https://tools.ietf.org/html/rfc1952

[23] T. A. Welch, ''A technique for high-performance data compression,'' *Computer*, vol. 17, no. 6, pp. 8–19, Jun. 1984.

[24] TransAID. (Jun. 2019). *System Prototype Demonstration*. Public Deliverable D7.2. [Online]. Available: https://www.transaid.eu/deliverables/

[25] *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service*, Standard ETSI EN 302 637-2 V1.3.2, ETSI TC ITS, Nov. 2014.

[26] *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Analysis of the Collective Perception Service (CPS)*, Standard ETSI TR 103 562 V2.1.1, ETSI TC ITS, Dec. 2019.

[27] *Intelligent Transport Systems (ITS); Vehicular Communications; Informative Report for the Maneuver Coordination Service*, Standard ETSI TR 103 578 V0.0.4, ETSI TC ITS, 2019.

[28] R. Molina-Masegosa, M. Sepulcre, J. Gozalvez, F. Berens, and V. Martinez, ''Empirical models for the realistic generation of cooperative awareness messages in vehicular networks,'' *IEEE Trans. Veh. Technol.*, vol. 69, no. 5, pp. 5713–5717, May 2020.

[29] TransAID. (Aug. 2019). *Definition of V2X Message Sets*. Public Deliverable D5.1. [Online]. Available: https://www.transaid.eu/deliverables/

[30] J. Schindler, D. Wesemeyer, A. Leich, M. Rondinone, and T. Walter, ''Cooperative intelligent transport systems: Towards high-level automated driving,'' in *Cooperative System Integration*. U.K.: IET Digital Library, 2019, pp. 257–283, doi: 10.1049/PBTR025E_ch12

[31] *Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Definitions*, Standard ETSI TR 102 638 V1.1.1, ETSI TC ITS, Jun. 2009.

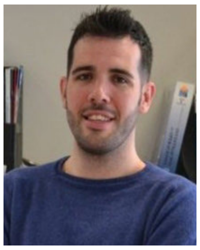
[32] G. Bansal, J. B. Kenney, and C. E. Rohrs, ''LIMERIC: A linear adaptive message rate algorithm for DSRC congestion control,'' *IEEE Trans. Veh. Technol.*, vol. 62, no. 9, pp. 4182–4197, Nov. 2013.

[33] *Technical Specification Group Radio Access Network; 'Study on LTE-Based V2X Services*, Standard TR 36.885 V14.0.0, 3rd Generation Partnership Project (3GPP), 2016.

MIGUEL SEPULCRE (Senior Member, IEEE) received degree in telecommunications engineering and the Ph.D. degree in communications technologies from the Universidad Miguel Hernández de Elche (UMH), Spain, in 2004 and 2010, respectively. He was a Visiting Researcher with ESA, Noordwijk, The Netherlands, in 2004, with the Karlsruhe Institute of Technology, Germany, in 2009, and with the Toyota InfoTechnology Center, Tokyo, Japan, in 2014. He is currently an Associate Professor with the Communications Engineering Department, UMH. He was awarded by the COIT (Spanish official association of Telecommunication Engineers) with the ONO Prize to the best Ph.D. thesis. He was the TPC Co-Chair of IEEE VTC2018-Fall, IEEE/IFIP WONS 2018, and IEEE VNC 2016. He serves as an Associate Editor for *IEEE Vehicular Technology Magazine* and IEEE COMMUNICATIONS LETTERS.

JAVIER GOZALVEZ (Senior Member, IEEE) received the degree in electronics engineering from the Engineering School ENSEIRB, Bordeaux, France, and the Ph.D. degree in mobile communications from the University of Strathclyde, Glasgow, U.K. Since October 2002, he has been with the Universidad Miguel Hernández de Elche (UMH), Spain, where he is currently a Full Professor and the Director of the UWICORE Laboratory. At UWICORE, he also leads research activities in the areas of vehicular networks, 5G and beyond networks, and industrial wireless networks. He has been an elected member to the Board of Governors of the IEEE Vehicular Technology Society (VTS) since 2011, and served as the President of the IEEE VTS in 2016 and 2017. He was an IEEE Distinguished Lecturer of the IEEE VTS, and served also as an IEEE Distinguished Speaker. He is the Editor-in-Chief of the *IEEE Vehicular Technology Magazine*. He was the General Co-Chair and the Founder of the IEEE Connected and Automated Vehicles Symposium 2018 and 2019, and the General Co-Chair of the IEEE VTC-Spring 2015 conference.

GOKULNATH THANDAVARAYAN (Graduate Student Member, IEEE) received the master's degree in computer science engineering from Anna University, India, in 2012. He is currently pursuing the Ph.D. degree with the Universidad Miguel Hernández de Elche (UMH), Spain. In 2015, he joined as a Researcher with United Arab Emirates University (UAE), and worked in different projects related to vehicular networks. In 2018, he joined UWICORE Research Laboratory to work in the European project (TransAID). His work focuses on improving the collective perception or cooperative sensing for the connected and automated vehicles using V2X communications.



BALDOMERO COLL-PERALES (Member, IEEE) received the M.Sc.Eng. and Ph.D. degrees (Hons.) from the Universidad Miguel Hernandez (UMH), Elche, Spain. He was formerly Postdoctoral Associate with HMETC (Russelsheim, DE), IIT-CNR (Pisa, IT), and WINLAB (Rutgers University, Brunswick, NJ, USA). He is currently a Research Fellow with the UWICORE Laboratory. His research interests include connected automated vehicles and device-centric technologies. He has served as a member for the TPC in more than 30 international conferences. He has served as the Track Co-Chair for IEEE VTC-Fall 2018. He is an Associate Editor of *Telecommunication Systems* (Springer) and *International Journal of Sensor Networks*.

JULIAN SCHINDLER received the Diploma in computer science from the Technical University of Braunschweig, Germany, in 2006. He directly started his career at the German Aerospace Center (DLR), Institute of Transportation Systems (ITS). After working on several national and international projects in the areas of driving simulation, ergonomics, and human centered design of cooperative vehicle automation systems, he became co-coordinator of the H2020 project MAVEN. Being the Leader of the Vehicle Automation Group, DLR-ITS, since 2017, he became the Coordinator of the H2020 Project TransAID.

MICHELE RONDINONE received the M.Sc. degree in telecommunications engineering from the University of Bologna, Italy, and the Ph.D. degree in industrial and telecommunications technologies from UMH. Since 2014, he has been with the Hyundai Motor Europe Technical Center, Germany, where he currently works as a Senior Engineer on V2X deployment preparation and future V2X for automated driving in internal as well as EU-funded projects. He actively participates in European V2X standardization at ETSI TC ITS and C2C-CC and served as the TCP Co-Chair for the IEEE Connected and Automated Vehicles Symposium 2018 as well as a TPC member for several IEEE international conferences.