AutoC2X: Open-source software to realize V2X cooperative perception among autonomous vehicles

Manabu Tsukada, Takaharu Oi, Akihide Ito, Mai Hirata, Hiroshi Esaki

AutoC2X: Open-source software to realize V2X cooperative perception among autonomous vehicles

Manabu Tsukada*, Takaharu Oi*, Akihide Ito*, Mai Hirata*,and Hiroshi Esaki*
* Graduate School of Information Science and Technology, The University of Tokyo, Tokyo, Japan
Email: {*tsukada, oi, akky, mai*} @*hongo.wide.ad.jp, hiroshi@wide.ad.jp*

***Abstract*—The realization of vehicle-to-everything (V2X) communication enhances the capabilities of autonomous vehicles in terms of safety efficiency and comfort. In particular, sensor data sharing, known as cooperative perception, is a crucial technique to accommodate vulnerable road users in a cooperative intelligent transport system (ITS). In this regard, open-source software plays a significant role in prototyping, validation, and deployment. Specifically, in the developer community, Autoware is a popular open-source software for self-driving vehicles, and OpenC2X is an open-source experimental and prototyping platform for cooperative ITS. This paper reports on a system named AutoC2X to enable cooperative perception by using OpenC2X for Autoware-based autonomous vehicles. The developed system is evaluated by conducting field experiments involving real hardware. The results demonstrate that AutoC2X can deliver the cooperative perception message within 100 ms in the worst case.**

***Index Terms*—Cooperative ITS, V2X, Cooperative perception, Autonomous vehicle, Open-source software**

I. Introduction

Presently, autonomous vehicles represent one of the essential development targets in the next-generation mobility service domain. However, the development of autonomous vehicles encounters several technical challenges in the domain of cyber-physical systems such as sensing, computing, and actuation. The sensing devices typically include LiDAR systems, cameras, and global navigation system satellite (GNSS)/ inertial measurement units. Using the sensor data, an autonomous vehicle can compute information pertaining to localization, detection, prediction, planning, and control. The result of this computation is sent to the controller area network (CAN) bus to control the vehicle. In this regard, several open-source software such as Autoware [1], [2] and Baidu Apollo [3] provide broad software packages and libraries necessary for autonomous vehicles.

Communication is another critical aspect of cyber-physical systems. Vehicle-to-everything (V2X) communications support vehicles to enable safer, more efficient, and more comfortable self-driving. In this regard, a cooperative intelligent transport system (ITS) represents a set of standards for the ecosystem of ITS applications among public-private entities. The common architecture, known as the ITS station architecture [4], [5] , was developed by the European Telecommunications Standards Institute (ETSI) and International Organization for Standardization (ISO). According to this standard, a cooperative awareness message (CAM) [6] is the most basic safety-related V2V message, using which, a vehicle can intimate the surrounding vehicles regarding its presence in real-time. A local database known as the local dynamic map (LDM) [7] stores the received presence information and provides support for various ITS applications. Many products in the market support these standards. Specifically, open-source systems such as OpenC2X [8] and Vanetza [9] offer software packages for these standards.

However, a cooperative ITS needs to accommodate vulnerable road users (VRUs, *e.g.,* pedestrians and bicycles) and legacy vehicles, neither of which are usually equipped with V2X sender devices. Using the concept of cooperative perception (also known as collective or collaborative perception), the presence information of these objects, as detected by local sensors mounted on the vehicle or roadside units, can be shared. ETSI is currently developing the standard for collective perception messages (CPMs) [10], [11].

Contribution: Considering this background, the objective of this study is to realize the interconnection of Autoware-based standalone autonomous vehicles under a cooperative ITS framework using OpenC2X. Cooperative perception is realized by sharing the information of the detected objects (vehicle, pedestrian, etc.) by Autoware among the neighboring nodes, taking into account the ETSI cooperative ITS standards. To the best of our knowledge, this is the first work to realize cooperative perception by integrating Autoware and OpenC2X. The proposed software, AutoC2X, is evaluated both in indoor and field tests, and the source code is available at *https://github.com/esakilab/AutoC2X-AW*.

The remaining paper is organized as follows. Section II provides a review of the related work. Section III describes the system design for integrating Autoware and OpenC2X. Section IV discusses the implementation of AutoC2X. Section V describes the evaluation setup and presents the results of the indoor and field tests. Section VI presents the conclusions and scope of future work.

II. Related Work

Sensor data sharing can be classified into three categories depending on the instant of occurrence of sensor fusion [12]: 1) In the raw/low-level method, the raw data from the sensors is shared. Such data include point clouds from LiDAR and camera images. 2) In the feature/middle-level method, pre-processed data such as a bounding box from a vision-based object detector are shared. 3) In the object/track-level method, the position information of the objects in the global coordinate

system is shared. Augmented vehicular reality (AVR) [13] systems share the raw-level sensor data (point cloud) of 3D camera depth perception sensors. Specifically, in [13], the authors described the motion prediction of dynamic objects to hide the latency, and off-the-shelf wireless technologies were considered to enhance the feasibility of the prototype system. In [14], the authors discussed the use of feature-level sensor data sharing to address the limited network bandwidth and stringent real-time constraints.

Some researchers also evaluated the realization of feature-level sensor data sharing in real vehicles [15]. Furthermore, object(track)-level cooperative perception has been realized using infrastructure involving multiple sensors and multiple transmitters [16], [17]. In [18], the researchers attempted to examine the trustworthiness of cooperative perception by quantifying the confidence in the correctness of data by using the Bayes theory. Furthermore, in [19], the authors tested the V2X cooperative perception and its application to ITS by considering the basic safety message (BSM) and conducting a field test involving three vehicles.

In our previous work, we proposed the use of a proxy CAM [20], [21] mechanism, in which a roadside sensor estimates the detected object location information, and a roadside transmitter broadcasts this information in the CAM format on behalf of the detected object. The receiver of the proxy CAM can process the message using the same reception procedure as that of the CAM. Consequently, the solution is compatible with all the products currently available in the market.

The cooperative automation research mobility application (CARMA) platform [22] is an open-source software that connects Autoware-based autonomous vehicles by using US standards [23], including BSM, Signal Phase and Timing (SPaT), and MAP. Although some maneuver coordination messages are defined in CARMA, cooperative perception (*i.e.* sensor data sharing) is currently not supported.

III. System Design

A. System Model

The system is designed by integrating two open-source software (Autoware and OpenC2X) to realize cooperative perception. Figure 1 shows the proposed system model. Instead of installing both the software in a computer, the system is designed such that a host realizes autonomous driving, and a router is in charge of the cooperative ITS function. This is because a vehicle often has a group of dedicated computers that requires external connectivity, such as driving log, mapping, and navigation. Thus, rather than extending all the nodes to have external connectivity, including the antenna, it is ideal to have a router in charge of managing the external connectivity for all the nodes.

This system model can be applied to a roadside unit (RSU) because it involves many nodes that require connectivity to the vehicles such as edge servers, traffic monitoring systems, signage, and signals. The system model of the RSU is a subset of Figure 1 as a certain functionality of autonomous driving is absent, *e.g.*, localization, planning, and CAN control.

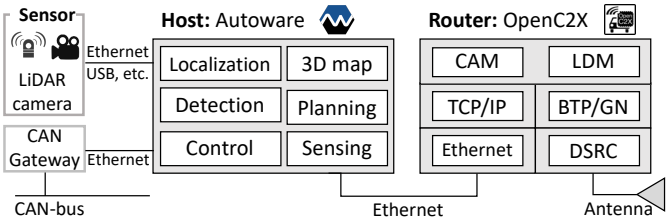


Fig. 1. System model to integrate Autoware and OpenC2X

B. Autoware

Autoware is based on the robot operating system (ROS) [24], which is an open-source middleware framework, which is widely utilized for robot application development. ROS is a distributed computing platform involving *nodes* and *topics*. The *nodes* represent the processing module of tasks, and the nodes communicate with one another via *topics*. Furthermore, ROS provides a powerful tool named *Rosbag* for recording and replaying the messages in *topics*. Using *Rosbag*, the developer can record the sensor data in the actual environment and later improve the algorithm by using the data without the hardware. Moreover, ROS includes a 3D visualization tool named *RViz*, which efficiently presents the status of the tasks. Autoware takes advantage of the abovementioned framework and integrated tools.

In addition, **3D maps** represents a common digital infrastructure for the operation of an autonomous vehicle, especially in urban areas. Autoware uses two types of 3D maps recorded in advances, such as vector map data and point cloud map data. The former is used to obtain the lane data, and the latter is used for scan matching to enable localization. **Sensing** in Autoware mainly involves the use of 360-degree LiDAR scanners and cameras. Point cloud data from the LiDAR scanners is used for the localization as well as the detection of the surrounding objects. The **localization** algorithm in Autoware employs scan matching between the 3D point cloud maps and the point cloud from LiDAR scanners. In general, the normal distributions transform algorithm [25] is employed for the localization. **Object detection** is performed based on the clustering of point cloud data from LiDAR scanners. In Autoware, the Euclidean cluster extraction algorithm is used [26]. After the clustering, Autoware calculates the distance between the detected objects and ego vehicle. Autoware provides software packages for prediction, decision, and planning (mission and motion planning); however, these are beyond the region of interest of this work and thus not considered. The **actuation** of autonomous vehicles is performed via a CAN controller to manipulate acceleration, braking, and steering.

C. OpenC2X

OpenC2X covers almost the entire protocol stack in the ITS station architecture, except for the security entity. In the access layer, the system supports Outside the Context of a BSS (OCB) mode for IEEE 802.11p and considers the decentralized con-

gestion control to adapt to the protocol behavior based upon the change in the vehicle density. In the transport and network layer, the system partially supports the basic transport protocol (BTP) [27] and GeoNetworking (GN) [28]. Specifically, it does not handle the packet forwarding, but the BTP and GN headers are added to the sending packets. In the facilities layer, CAM, LDM, and the decentralized environmental notification message (DENM) [29] are incorporated. CAMs are triggered periodically from 100 to 1000 ms, according to the standards. Furthermore, a user can trigger the DENMs from the web interface or using connected ITS applications, *e.g.*, collision avoidance applications. The LDM stores all the contents from the received CAM and DENM. In addition, OpenC2X provides a web-based graphical user interface to visualize the status of the ITS station.

IV. AutoC2X

A. Implementation Overview

Figure 2 shows the overview of AutoC2X implementation. AutoC2X is developed using the C++ language, by extending both Autoware (v. 1.11.1) and OpenC2X (standalone v. 1.5). The boxes in the figure show the functions; the arrows from left to right (→) and right to left (←) represent the sender side sequence and receiver side, respectively.

When the system is initiated, Autoware reads the 3D maps and starts receiving the sensor data. Using the point cloud data from the LiDAR, the system localizes the ego vehicle and detects the neighboring objects. The coordinates of the ego vehicle and the detected objects are transformed from the local coordinates system to the global coordinates system and sent to the OpenC2X router with TCP/IP over the Ethernet. OpenC2X encodes the ego vehicle coordinates in the CAM and neighboring objects coordinates in the proxy CAM form and transmit the information over dedicated short range communication (DSRC) with the BTP/GN header. The LDM simultaneously stores the information.

When the OpenC2X router receives CAMs and proxy CAMs, it extracts the information of the neighboring objects from the message and stores this information in the LDM. Then, OpenC2X sends the coordinates for the objects to Autoware. Finally, the data of the objects derived from the V2X communication is visualized in *RViz* after the coordinate transformation. The data regarding the decision and planning function is not fed back to the Autoware. Realizing this feedback is a part of future work.

B. Localization

Autoware and OpenC2X employ scan matching and GPS for the localization, respectively. In other words, the localization function is duplicated in the host and the router in the ITS station. In the proposed system, the localization function of Autoware is employed in the entire ITS station, owing to the high localization accuracy of scan matching, which is typically less than 10 cm. Furthermore, scan matching is more robust in the urban environment compared to the GPS. Thus, in the developed system, in Autoware, the ego vehicle location

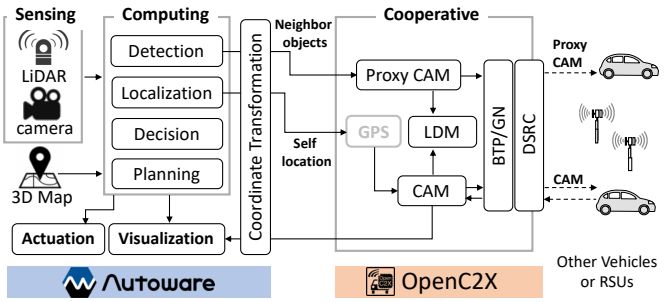


Fig. 2. Implementation overview

is published in the topic of `/ndt_pose`, and it is sent to OpenC2X after the coordinate transformation. OpenC2X uses the received position instead of the GPS.

C. Cooperative Perception

Autoware detects the surrounding objects by using a LiDAR and camera. After the sensor fusion, the information of the detected objects is published in the topic of `/detection/objects`. The proposed Autoware extension performs coordinate transformation and sends the information to OpenC2X. The proposed OpenC2X extension adds the received detected object information to a queue and encodes the latest information in the queue into a proxy CAM at a frequency of 10 Hz. The information currently available in the extension contains the timestamp, latitude, longitude, speed, and ITS station ID, and the other fields of CAM are left as "unknown". In the future, if additional sensors can be used to estimate more information, such as the vehicle length and heading, more fields can be filled out. The ITS station IDs of the detected objects are assigned random numbers.

D. Coordinate Transformation

Autoware maintains the coordinates by using the self-centered local coordinate system, and OpenC2X uses the geodetic reference system (i.e., latitude and longitude). Thus, a coordination transformation must be performed.

The proposed Autoware extension first transforms the local coordinate system into the universal transverse Mercator (UTM) coordinate system, according to the 3D map. The location of the detected object in the UTM coordinate system $\binom{X_{obj}}{Y_{obj}}$ can be expressed as

$$\begin{aligned}\begin{pmatrix} X_{obj} \\ Y_{obj} \end{pmatrix} &= \begin{pmatrix} X_{ego} \\ Y_{ego} \end{pmatrix} + R(\alpha) \begin{pmatrix} x_{obj} \\ y_{obj} \end{pmatrix} \\ &= \begin{pmatrix} X_{ego} \\ Y_{ego} \end{pmatrix} + \begin{pmatrix} \cos\alpha & -\sin\alpha \\ \sin\alpha & \cos\alpha \end{pmatrix} \begin{pmatrix} x_{obj} \\ y_{obj} \end{pmatrix} \\ &= \begin{pmatrix} X_{ego} + x_{obj}\cos\alpha - y_{obj}\sin\alpha \\ Y_{ego} + x_{obj}\sin\alpha + y_{obj}\cos\alpha \end{pmatrix}\end{aligned} \tag{1}$$

where $\binom{X_{ego}}{Y_{ego}}$ is the location of the ego vehicle in the UTM coordinate system, published in `/ndt_pose`; $R(\alpha)$

is the rotation of the ego vehicle with respect to the UTM coordinate system, published in `/tf`; and $\begin{pmatrix} x_{obj} \\ y_{obj} \end{pmatrix}$ is the relative position of the detected objects from the ego vehicle when the heading of the ego vehicle is in the X-axis, published in the topic of `/detection/objects`. Subsequently the extension transforms the UTM coordinate system into the geodetic reference system 1980 (GSR80) by using the PROJ library [30].

Conversely, when the extension receives a V2X message in the GSR80 form from OpenC2X, the extension transforms the coordinates into the local coordinate system to handle it in Autoware.

E. Visualization

The proposed Autoware extension publishes the detected object information from V2X to `/detection/objects`. *RViz* visualizes the objects in the topic, as shown in Figure 3. In the figure, an autonomous vehicle is approaching an intersection. The green boxes around the vehicle highlight the objects detected by the local LiDAR sensor, and the green squares near the intersection show the object received from the V2X cooperative perception. To facilitate the visualization, we currently use a square in which the latitude and longitude of the object are at the center. Such objects can also be visualized using other shapes, depending on the vehicle type and size, if the received CAM has such information.

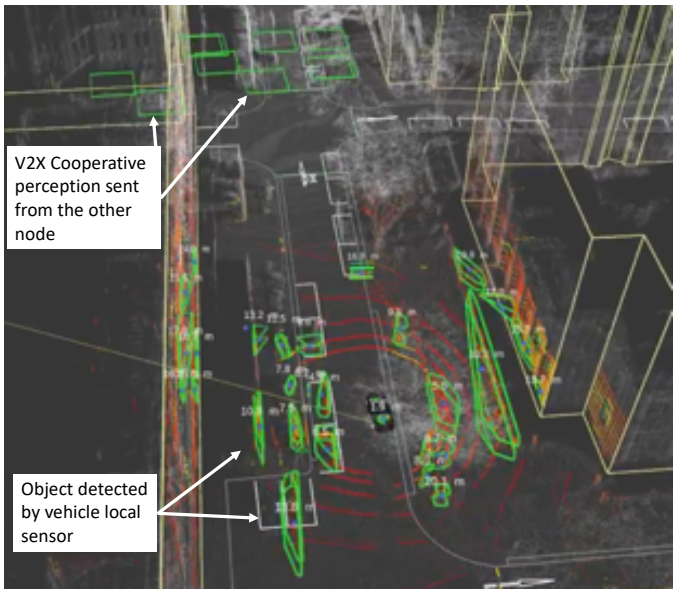


Fig. 3. Screenshot of RViz with AutoC2X

V. Evaluation

A. Experimental Setup and Evaluation Metrics

For the evaluation, we consider a scenario in which an RSU located at an intersection broadcasts the cooperative perception (proxy CAM) to the surrounding vehicles, as shown in Figure 4 (at Hongo campus, the University of Tokyo, Japan). Both the RSU and vehicle have the same set of equipment (an OpenC2X router, an Autoware host, an antenna, and a LiDAR sensor). The routers are APU4C4 embedded routers with an Intel Wi-Fi 6 AX200 module, on which Ubuntu 18.04.3 LTS is installed. The hosts are laptop PCs (CPU Core i7 8-cores) with Ubuntu 16.04.5 LTS. The RSU and vehicle hosts have a 16 GB and 32 GB memory, respectively. Both the RSU and the vehicle are equipped with Velodyne VLP-16 LiDAR sensors. According to the Japanese regulations, IEEE 802.11g ad-hoc mode is used for the tests.

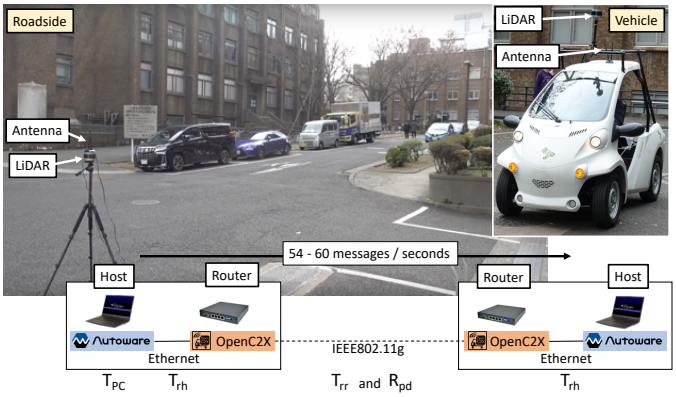


Fig. 4. Experimental setup

As shown in the bottom of Figure 4, the packet delivery ratio (PDR) R_{pd} and total delay T are considered as the evaluation metrics. R_{pd} is considered as the PDR between the routers because no packet loss is observed inside the ITS station (i.e., Ethernet link) in all the experiments. R_{pd} in one way was measured by comparing the LDM entries of the sender and receiver after the experiments. The total delay T is expressed as

$$T = T_{pc} + T_{hr} + T_{rr} + T_{rh} \tag{2}$$

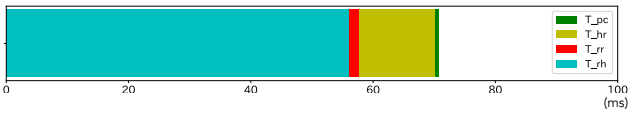
where T_{pc} is the processing delay in Autoware. T_{hr}, T_{rr}, and T_{rh} express the transmission delay from the host to the router, between routers, and from the router to the host, respectively. T_{pc} includes the processing delay in Autoware pertaining to the clustering of the point cloud, detection, and localization of the objects; this value is calculated considering the gap between the published time of the point cloud and the time of the message to the router. T_{hr}, T_{rr}, and T_{rh} are calculated considering half the round-trip time for each link. The proposed extensions for both Autoware and OpenC2X return the same message to the sender upon the reception of the message, to enable the measurement.

B. Indoor Test

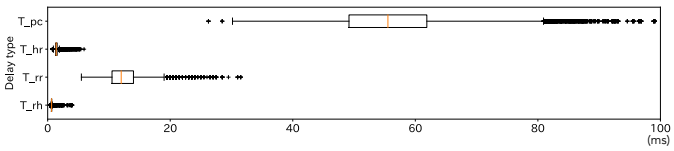
The indoor test was conducted by placing the routers and hosts, as shown in Figure 4, on a desk. The distance between the routers was approximately 3 m. The Rosbag for the RSU was recorded at an intersection at the Hongo campus, and that for the vehicle was recorded around the school. The Rosbags had a duration of 4 min 35 s, and they were replayed in the

RSU and vehicle hosts. The RSU generated an average of 57 messages per second, and the measurement was conducted five times.

Figure 5 shows the results. As in Figure 5(a), the average total delay T was 70.7 ms, of which T_{pc} occupied 80%. Figure 5(b) shows that the maximum delay pertaining to T_{pc}, T_{hr}, T_{rr}, and T_{rh} were approximately 80 ms, 3 ms, 20 ms and 3 ms, respectively. Therefore, the autonomous vehicle could receive the cooperative perception message within approximately 100 ms in the worst case. The frequency of point cloud measurement was 10 Hz, which was also the maximum frequency for the CAM. Therefore, it can be considered that the proposed system can deliver the message within the point cloud measurement interval, as well as within the CAM interval.



(a) Average total delay T



(b) Breakdown of delay (T_{pc}, T_{hr}, T_{rr} and T_{rh})

Fig. 5. Total delay and the breakdown

C. Field Experiment

R_{pd} was measured in the field experiments. Figure 6 shows the location of the RSU and the driving route of the vehicle. The vehicle drove along the yellow line in the figure at approximately 3 km/h several times within a total time of 1000 s. In addition, Figure 6 shows the experimental result. The color of the tile indicates the R_{pd} at the location. A lack of tiles indicates that no message was received at the location.

It can be seen that in most cases, R_{pd} gradually decreases as the distance increases. In addition, the building filters the messages, leading to a sharp decrease in the PDR value. The left and right sides of the RSU respectively correspond to an upslope and a downslope with many trees. Consequently, the left side has a better line of sight (LOS), leading to a better PDR. The result shows that the PDR is more than 80% when the vehicle is within 30 m and has a LOS to the RSU. The PDR degrades at the bottom of the dip located in the center of the figure.

VI. Conclusion and Future Work

In this work, a system named AutoC2X was developed to enable the cooperative ITS function using OpenC2X for Autoware-based autonomous vehicles. Furthermore, AutoC2X can realize V2X cooperative perception by using a proxy CAM. The developed system was evaluated in field experiments involving real hardware. The results indicated that

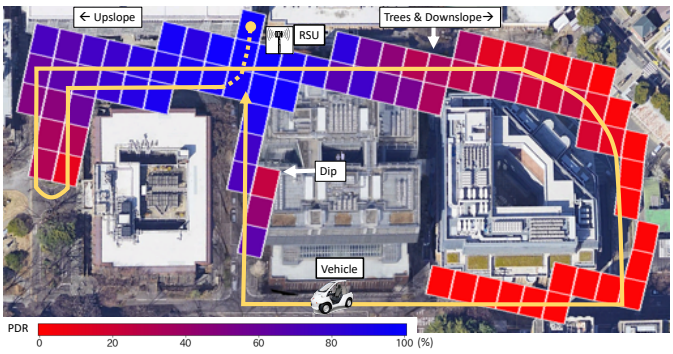


Fig. 6. Packet delivery ratio R_{pd} in field experiment

AutoC2X can deliver the cooperative perception message within 100 ms in the worst case.

Future work is expected to include field evaluations involving large-scale vehicular networks. In addition, more information of the detected objects, such as the vehicle length and heading can be obtained, and the data can be encoded to the proxy CAM. In this work, the received object data via V2X were only visualized; however, we intend to use the data to realize additional autonomous vehicle functions, such as decision and planning. Furthermore, once the standard for a CPM is established, the proxy CAM module can be replaced by the CPM module.

Acknowledgment

This work was partly supported by JSPS KAKENHI Grant Number JP17H04678.

References

[1] S Kato, E Takeuchi, Y Ishiguro, Y Ninomiya, K Takeda, and T Hamada. An open approach to autonomous vehicles. *IEEE Micro*, 35(6):60–68, November 2015.

[2] S Kato, S Tokunaga, Y Maruyama, S Maeda, M Hirabayashi, Y Kitsukawa, A Monrroy, T Ando, Y Fujii, and T Azumi. Autoware on board: Enabling autonomous vehicles with embedded systems. In *2018 ACM/IEEE 9th International Conference on Cyber-Physical Systems (ICCPS)*, pages 287–296, April 2018.

[3] Baidu Apollo. http://apollo.auto/.

[4] ISO 21217:2014 Intelligent transport systems – Communications access for land mobiles (CALM) – Architecture, April 2014.

[5] Intelligent Transport Systems (ITS); Communications Architecture, September 2010. ETSI EN 302 665 V1.1.1 (2010-09).

[6] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 2: Specification of Cooperative Awareness Basic Service, 2019. ETSI EN 302 637-2 V1.4.1 (2019-04).

[7] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Local Dynamic Map (LDM), September 2014. ETSI EN 302 895 V1.1.1 (2014-09).

[8] S Laux, G S Pannu, S Schneider, J Tiemann, F Klingler, C Sommer, and F Dressler. Demo: OpenC2X — an open source experimental and prototyping platform supporting ETSI ITS-G5. In *2016 IEEE Vehicular Networking Conference (VNC)*, pages 1–2, December 2016.

[9] Raphael Riebl, Christina Obermaier, Stefan Neumeier, and Christian Facchi. Vanetza: Boosting research on inter-vehicle communication. *Proceedings of the 5th GI/ITG KuVS Fachgespräch Inter-Vehicle Communication (FG-IVC 2017)*, pages 37–40, 2017.

[10] Intelligent Transport Systems (ITS); Cooperative Perception Services (CPS), December 2019. ETSI TS 103 324 V0.0.14 (2019-10).

[11] Intelligent Transport Systems (ITS);Vehicular Communications; Basic Set of Applications; Analysis of the Collective Perception Service (CPS); Release 2 , December 2019. ETSI TR 103 562 V2.1.1 (2019-12).

[12] Michael Aeberhard and Nico Kaempchen. High-level sensor data fusion architecture for vehicle surround environment perception. In *Proc. 8th Int. Workshop Intell. Transp*, 2011.

[13] Hang Qiu, Fawad Ahmad, Fan Bai, Marco Gruteser, and Ramesh Govindan. AVR: Augmented vehicular reality. In *Proceedings of the 16th Annual International Conference on Mobile Systems, Applications, and Services*, pages 81–95. ACM, June 2018.

[14] Qi Chen. F-Cooper: Feature based cooperative perception for autonomous vehicle edge computing system using 3D point clouds. September 2019.

[15] Akihiro Nakamura, Shun Taguchi, Yoshihiro Ohama, Mitsuhiro Araki, and Kunihiro Goto. Multiple-Object tracking with cameras mounted on vehicles for generating a local dynamic map. In *Proceedings of the 5th International Symposium on Future Active Safety Technology toward Zero Accidents (FAST-zero '19)*, September 2019.

[16] Eduardo Arnold, Mehrdad Dianati, and Robert de Temple. Cooperative perception for 3D object detection in driving scenarios using infrastructure sensors. December 2019.

[17] M Gabb, H Digel, T Müller, and R Henn. Infrastructure-supported perception and track-level fusion using edge computing. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1739–1745, June 2019.

[18] Christoph Allig, Tim Leinmuller, Prachi Mittal, and Gerd Wanielik. Trustworthiness estimation of entities within collective perception. In *2019 IEEE Vehicular Networking Conference (VNC)*, 2019.

[19] R Miucic, A Sheikh, Z Medenica, and R Kunde. V2X Applications Using Collaborative Perception. In *2018 IEEE 88th Vehicular Technology Conference (VTC-Fall)*, pages 1–6, August 2018.

[20] T. Kitazato, M. Tsukada, H. Ochiai, and H. Esaki. Proxy cooperative awareness message: an infrastructure-assisted v2v messaging. In *2016 Ninth International Conference on Mobile Computing and Ubiquitous Networking (ICMU)*, pages 1–6, Oct 2016.

[21] Manabu Tsukada, Masahiro Kitazawa, Takaharu Oi, Hideya Ochiai, and Hiroshi Esaki. Cooperative awareness using roadside unit networks in mixed traffic. In *2019 IEEE Vehicular Networking Conference (VNC)*, pages 9–16, 2019.

[22] FHWA Saxton Transportation Operations Laboratory. Architecture for CARMA System Version 3.0. Technical report, February 2020.

[23] *Dedicated Short Range Communications (DSRC) Message Set Dictionary*, March 2016. SAE International J2735.

[24] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. ROS: an open-source robot operating system. In *ICRA workshop on open source software*, volume 3, page 5, 2009.

[25] P Biber and W Strasser. The normal distributions transform: a new approach to laser scan matching. In *Proceedings 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003) (Cat. No.03CH37453)*, volume 3, pages 2743–2748 vol.3, October 2003.

[26] Radu Bogdan Rusu. Semantic 3D object maps for everyday manipulation in human living environments. *KI - Künstliche Intelligenz*, 24(4):345–348, November 2010.

[27] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 5; Sub-part 1, August 2014.

[28] Intelligent Transport Systems (ITS); Vehicular Communications; GeoNetworking; Part 4: Geographical addressing and forwarding for point-to-point and point-to-multipoint communications; Sub-part 1: Media-Independent Functionality, May 2019.

[29] Intelligent Transport Systems (ITS); Vehicular Communications; Basic Set of Applications; Part 3: Specifications of Decentralized Environmental Notification Basic Service, November 2014. ETSI EN 302 637-3 V1.2.2 (2014-11).

[30] PROJ contributors. *PROJ coordinate transformation software library*. Open Source Geospatial Foundation, 2020.