Multimedia Tools and Applications (2022) 81:12349–12370
https://doi.org/10.1007/s11042-021-11421-x

1172: 5G MULTIMEDIA COMMUNICATIONS FOR VEHICULAR, INDUSTRY AND ENTERTAINMENT APPLICATIONS

5G MEC-enabled vehicle discovery service for streaming-based CAM applications

Gorka Velez[1] · **Josu Perez**[1] · **Angel Martin**[1]

Received: 22 June 2020 / Revised: 9 February 2021 / Accepted: 10 August 2021 /
Published online: 17 September 2021

Abstract

Cooperative perception represents an important technology to fulfil the higher automation levels of connected and automated mobility (CAM). In cooperative perception, the sensor data, either raw or processed, is shared among neighbour vehicles with the objective of enhancing or complementing the perception obtained by on-board sensors. The vehicle that requests this external perception data needs to have this data quickly. However, it first needs to discover the network address of the neighbour vehicle that wants to connect to. Specially in a dense urban area or in a congested radio channel, an inefficient method for neighbour vehicle discovery could prevent a timely start of the cooperative perception session. This paper describes a novel 5G multi-access edge computing (MEC) solution that that boosts the selection of interesting neighbour vehicles according to a geographical region of interest (ROI) after applying pertinent adjustments considering vehicles' dynamics and network communication latencies. In contrast to broadcast-based methods, in the proposed method the vehicles are only sending their periodical position data to a MEC service, which centralises the vehicle discovery requests. The objective of this Vehicle Discovery Service (VDS) is to support the startup of Web Real-Time Communications (WebRTC)-based Extended Sensors CAM applications. The proposed VDS has been validated using a public vehicular traffic dataset evaluating geo-position accuracy. The WebRTC-based streaming pipeline has been validated testing its feasibility for a See-Through video streaming application.

Keywords 5G · MEC · Streaming · Vehicle Discovery Service (VDS) · WebRTC

✉ Gorka Velez
gvelez@vicomtech.org

Josu Perez
jperez@vicomtech.org

Angel Martin
amartin@vicomtech.org

[1] Vicomtech Foundation, Basque Research and Technology Alliance (BRTA), Mikeletegi 57, 20009 Donostia-San Sebastián, Spain

1 Introduction

The growing amount of available car data brings new opportunities around the automotive industry. Hence, the exploitation of car data is strategic to achieve revenue generation, cost reduction and enhancement of safety and security. Thus, data-driven autonomous driving applications and smart mobility services will change traditional automotive value chain with a market volume equivalent to vehicle sales by 2030 [29].

Specifically, autonomous driving systems will transform the mobility paradigm in many ways, from safety to energy efficiency. Industry players are focused on different value creation models such as: revenues generation by data-based selling/advertising of products or reselling data to third parties; costs reduction by gathering product field data to reduce research and development costs for optimizing materials and designs; and increased safety and security by reducing time for intervention of first responders and triggering forward warnings.

Cellular Vehicle-to-Everything (C-V2X) is the main approach from communication standard bodies to bridge communications between vehicles and Connected and Automated Mobility (CAM) service infrastructures [7]. Initially defined in Long Term Evolution (LTE) Release 14 [11] and evolved along present and future 5G releases, European Telecommunications Standards Institute (ETSI) and Third Generation Partnership Project (3GPP) define two main communication modes for vehicles: direct Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I). Both communication modes use sidelink interfaces and licensed cellular bands but they bypass the Base Station (BS). V2V and V2I are added to the regular Vehicle-to-Network (V2N) mode as any other User Equipment (UE) subscribed to a BS.

As the vehicle's connectivity turns into versatile and universal, the natural evolution of autonomous driving systems requires going beyond the limitations of on-board sensors. This vision will grant access to data coming from other vehicles to support the generation of a more holistic understanding of the environment. External data flows must come with a pace and latency rates similar to on-board systems to avoid synchronisation inconsistencies when fusing multi-origin data. So, a major challenge is related to the communications latency as systems must reliably transmit under tight latency conditions [21]. CAM applications requires quick, effective and secure communication among vehicles and service infrastructures.

Multi-access edge computing (MEC) brings close-to-zero latency and scalable processing while enforces privacy limiting data range to a local environment [31]. Edge infrastructures turn into key actors when it is required to process published metadata, to retrieve events or actionable data, or to boost trusted discovery and handshake. Furthermore, Mobile Network Operators (MNOs) foresee a big opportunity to create new revenues flows from hosting third-party service components at edge infrastructures.

In order to exploit the available data, the CAM services need to first identify the relevant data sources for each vehicle. The geo-position of each data source is an essential information to effectively choose the most appropriate data source for a specific vehicle. However, the required low latency, high reliability and trust in the external information from all the available data sources mean a paramount challenge under some vehicle density conditions. Consequently, the vehicle might not be able to timely obtain the network address of the data source that suits best its purposes.

In traditional V2V approaches each vehicle transmits its perception data to the rest of the vehicles. However, transmitting such an amount of messages causes a high load on the radio channel, especially in dense urban environments with a large number of vehicles. The

high network load could in turn cause packet losses and a large communication latency, degrading the accuracy and timeliness of the perception messages [17]. In our approach, we concentrate on the design and implementation of a MEC service, which boosts the discovery and selection of surrounding sensor data sources for video-based CAM services, such as See-Through [34] or extended perception. Here, low start-up time and latency are essential as the on-board Advanced driver-assistance systems (ADAS) systems will require under frame-level delay to consistently fuse local captured and external received video flows. Furthermore, the selection of an appropriate sensor data source is not easy to conclude as it involves multiple vehicles, moving with different directions, speeds and accelerations, and a one-by-one or broadcast query could mean a late reaction. In this context, a vehicle discovery service is used to select an external data source from the surrounding sensor data sources, previously registered as streaming services.

The proposed 5G MEC-enabled vehicle discovery service centralises all the geo-tagged data in a local area, to efficiently select the best vehicle candidates from the ones published in the surrounding area. It stores recent geo-position history to suggest a list of sensor data sources, provided as streaming services, matching the upcoming geo-positions of the vehicles. To this end, the vehicle discovery algorithm considers the vehicles dynamics and the network performance to estimate upcoming geo-position shifts, which could lead to a sub-optimal or even useless selection.

The major contributions of our work in order of appearance in this paper, can be summarized as follows:

- The architecture and implementation details of a novel 5G MEC-enabled vehicle discovery service for video streaming-based CAM applications such as See Through. The proposed implementation is built on top of industry standard technologies such as Message Queuing Telemetry Transport (MQTT) for IoT messaging or InfluxDB for efficiently storing and retrieving time-stamped data. In contrast with common cooperative perception approaches, in the proposed approach there is no broadcast messaging but a centralised MEC service that gathers the required data to respond to vehicle discovery queries. The proposed vehicle discovery service employs a lightweight algorithm that considers vehicle dynamics and network latency to select the appropriate vehicles as sensor data sources.
- A See-Through CAM application implementation using Web real-time communications (WebRTC) for secured, NAT-transversal and low-latency video streaming.
- The analytical decomposition of the proposed See-Through CAM application's start-up and delay times into operational ranges and pipeline steps to make a deeper end-to-end analysis beyond theoretical 5G communications Key Performance Indicators (KPIs) for identified target use cases.
- The analysis of spatial accuracy of the proposed vehicle discovery method when forecasting the geo-position of vehicles once the selection is received by the vehicle which is performing the query. The accuracy is evaluated in an intra-cell scenario, where one MEC server processes the request and serves the response as vehicles move.
- The evaluation of the end-to-end latency of a WebRTC-based See-Through CAM application according to the difference between captured and received video frame timestamps, including different operational settings in the pipeline.

The rest of this paper is organized as follows. First, the related work is in Section 2. The MEC system model and architecture and the ROI-based vehicle selection algorithm is in Section 3. Then, Section 4 presents the CAM service and its communication stack and

problem formulation. Furthermore, the system performance analysis, defined metrics, the evaluation setup and the simulation results are described in Section 5. Finally, the results are discussed in Sections 6 and 7 concludes the paper.

2 Related work

3GPP has identified a set of V2X scenarios grouped on five categories or use cases [2]: Advanced Driving spans complex manoeuvres such as, overtaking or cooperative collision avoidance; Vehicles Platooning aims effectively grouping a set vehicles to travel together one after the other; Remote Driving enables a remote driver to operate a vehicle; Quality of Service (QoS) Support, considered horizontal, to enforce steady network QoS; and Extended Sensors implies extending the perception obtained by the on-board sensors with sensor data received from surrounding vehicles or roadside units (RSUs). In all of them a poor performance of the communication pipeline, in terms of latency, may lead to a decrease in the level of automation and risks for reliable safety-critical use cases. Specifically, we focus on Extended Sensors as it brings a common initial step for different use case categories to establish communication across the participants in a use case.

The specification of C-V2X, originally declared in 3GPP Release 14 to empower LTE technologies, has evolved in Release 15 to gain the benefits from 5G New Radio (NR) such as improved reliability, increased capacity and reduced latency [13]. Then, further enhancements will come in Release 16 and subsequent Releases [14]. Beyond the 5G NR features, 5G brings a wider set of possibilities.

First, techniques to deploy a dynamic radio setup by means of Multiple-input Multiple-output (MIMO) antennas to enforce coverage areas according to UEs densities and geographic distributions [5]. Second, the application of Virtual Network Functions (VNF) and Software Defined Network (SDN) technologies, present in the ETSI architecture of 5G networks, to support the vision of the Self-Organised Network (SON) [20] empowered by machine learning to dynamically allocate network resources [6] or adapt network topology [26] based on traffic prediction and adaptable physical layer settings.

Different research activities target the reduction of the latency of V2V communications [3]. However, a common conclusion is that V2V communications scheme for fully distributed peers means low efficiency, high overheads and increased latency as the number of surrounding vehicles scales up [17]. Here, the third research pillar, the MEC services [10], can make the difference for time-critical applications as the number of subscribers increases [19]. MEC services exploit close-to-zero latency and distributed position to perform analytical functions by means of cloud services which are closer to the users. So, MEC means an enabler for enhanced C-V2X communication with a centralised system which manages a local area with more efficient and lower latency response. This way, CAM services operated in MEC infrastructures as a roadside unit can send information to nearby cars with extremely low latency, enabling instant reaction of ADAS systems, while limiting data transmission inside the cell and preserving data privacy. Furthermore, 5G MEC architecture can perform video analytics to enhance and boost live video streaming applications [27]. Moreover, MEC approaches has also been studied to support V2V communications while reducing latency in the context of multiple operators for roaming situations [28, 35]. Our approach goes a step further, moving from a V2V communications scheme to a V2I approach, where a MEC service, based on vehicle's ROI, provides the best data stream candidates according to their geo-position and dynamics.

In order to consume an edge service, the first step is to discover its presence and available functions and services. To this end, different service discovery protocols such as Universal Plug and Play (UPnP) or Simple Service Discovery Protocol (SSDP) can be used, with strict limitations to local networks as they are based on UDP or multicast communications. Beyond the local network domain, Extensible Messaging and Presence Protocol (XMPP) has been widely employed by multimedia web services for messaging and presence. Then, coming from the semantic web Universal Description, Discovery, and Integration (UDDI) and Web Services Description Language (WSDL) tandem facilitates services discovery and introspection. Other approaches push discovery solutions down from the application layer to the IP layer meaning integration of advanced features in the network stack of all the participants. Some are based on DNS [38]. A vehicular protocol from IP layer is the Vehicular Neighbor Discovery (VND) for IP-Based Vehicular Networks which implies IP header extensions [18]. However, the scalability for a distributed network of geographically distributed edge services is not managed by design by previous technologies.

In vehicular ad hoc networks (VANETs), wireless networks are spontaneously created from mobile devices. In this context, neighbor discovery refers to a mechanism that keeps each vehicle aware of its active neighbor vehicles. Even if different approaches exist to improve the accuracy of neighbor discovery [25], they are all essentially based on broadcast messages. The main drawback of these methods is that the radio channel can be overloaded with the periodical broadcast messages. Moreover, as these methods are designed for VANETs, they are only suited for short-range radio technologies and not for cellular technologies like LTE or 5G.

An essential parameter to perform selection of sensor data source from the surrounding cars is the geo-position accuracy, to be sure that the provided information will be relevant and complement the data captured from on-board sensors. This aspect has been also studied in the context of collective perception or cooperative sensing for vehicles and infrastructure nodes [33] analysing the communications overhead and the geo-position accuracy trade-off. Other works explore the possibility to anticipate or forecast values when exchanging data in a V2V schema [17] for a heavy network load. This tries to reduce communications overheads of V2V to reduce the negative impact on available bandwidth, which is low on V2V communications, and on latency, which may increase from sequenced V2V messages. Instead, our MEC approach makes a low latency, synchronous and coordinated anticipation of geo-positions shifts to provide a relevant sensor data source considering elapsed time for processing and transmission.

The solution proposed in this paper considers not only the geo-position of the available sensor data sources provided as streaming services, but also the trajectories and network delivery times to accurately estimate the data services relevant for the vehicle's trajectory. Thus, our approach focuses on the identification or a relevant surrounding data source applying adjustments on candidates based on trajectories and network performance to compensate delivery time of messages. This pushes the decision from the client-side to an edge system which can select better candidates from a better position exploiting all the data and stats on a local area. The discovery of the Vehicle Discovery Service endpoint is out of scope of the present work and would be based on application logic.

In terms of 5G network KPIs, both documents [2] and [1], declare a range of operational values which should satisfy data capacity, user density, transmission reliability and latency communications required by the different identified use cases. Specifically, for Extended Sensors category, those documents declare a transmission rate of 10 messages/s, a maximum end-to-end latency ranging from 10 to 100 ms, for a data rate of 10 to 90 Mbps, and a

minimum required communication range from 50 to 1000 meters. These theoretical values are ambitious and challenging as some field tests [16] of 5GNR communications with edge-computing features, in the 2.6 GHz frequency band delivering a 100 Mbps performance, achieve an average delay of 17 ms for vehicle-to-network-to-vehicle communications at a vehicle speed of 100 km/h.

3 Vehicle discovery service

The objective of using a MEC-centred approach is to reduce the amount of transmitted data without decreasing the application performance. The MEC centralises the geo-positioning data of the surrounding vehicles and offers a Vehicle Discovery Service (VDS), instead of having each vehicle sending constantly their geo-positioning data to the rest of the vehicles. In our approach, the MEC receives and digests the geo-positioning messages. If a vehicle is interested in gathering the geo-positioning data of the vehicles that are beyond its on-board sensors field of view, this can be done by simply querying this information to the MEC. So, a vehicle only receives requested information, and the communication channel is not overloaded with duplicated messages that most of the time are ignored by the receivers.

In a commercial deployment, the vehicle discovery service could act as a third-party service that is trusted by surrounding vehicles in order to identify and connect to neighbour vehicles and manage signalling and negotiation. The VDS acts as a local centralised solution to allow a more efficient and coordinated communication, instead of forcing a vehicle willing to communicate to query each neighbour vehicle one by one.

3.1 Architecture

The architecture of the proposed VDS is depicted in Fig. 1. The vehicles periodically send their geo-positioning data using a publish-subscribe network protocol. The MEC receives the vehicle data and stores it in a time series database (TSDB). TSDBs are optimised for storing and serving data through associated pairs of time and values. The objective of using a TSDB is to have an efficient tool to retrieve on demand the latest measurements received by the MEC. When a certain vehicle wants to gather the geo-positioning data of surrounding vehicles, it requests this information to the VDS deployed at the MEC. Then, this service queries the TSDB to retrieve the latest positions for each subscribed vehicle. In the implementation presented here, MQTT is proposed as the publish-subscribe network protocol and InfluxDB as the TSDB solution. MQTT is a lightweight protocol, widely used in Internet of Things (IoT) applications, that has open source client-server implementations such as Eclipse Mosquitto [23]. InfluxDB is also open source, and it is designed to handle high write and query loads.

It might not be efficient nor interesting for the requesting vehicle to receive the information about all the vehicles subscribed in the MEC. It is therefore necessary for the VDS to define a Region of Interest (ROI) in the area that it covers. The ROI includes only the vehicles that are inside an interesting range for the requesting vehicle. The VDS does not just forward the vehicle positions stored in the TSDB that are inside the ROI.

The ETSI Collective Perception Messages (CPM) [12], which are the standard messages for sending this kind of vehicle data, are sent with a frequency of 10 Hz. This means that the measurements stored in the TSDB can be up to 100 ms old. There is also some communications latency receiving the vehicle request and sending back the information about the vehicles in the ROI. These time offsets can mean significant values in the spatial domain

ruining the vehicle discovery result. To mitigate this problem, when a vehicle queries the VDS, the VDS transforms all the stored neighbour vehicle positions to the time instant when the response message is estimated to reach the vehicle that has queried the VDS. Once the neighbour vehicle positions are updated, the vehicles inside a ROI are included in the query result.

Three different MQTT topics are implemented to manage the messages. The first topic, $Feed$, is dedicated to feed the VDS with new vehicle geo-positioning measurements. The VDS is the only subscriber as it is the responsible of centralising all the data in the area. This is key to minimise the channel load. Then, a second topic, $Query$, is provided to the vehicles to query the VDS. The third topic, $Query\_result$, is employed by the vehicles to receive the query results from the MEC. This publish-subscriber protocol with three independent topics ensures an efficient use of the communication channel and minimises the irrelevant data received by the vehicles.

In the following subsection the method to generate the query result is described in more detail.

3.2 VDS response to a query

As explained in the previous subsection, the MEC cannot simply forward the latest measurements stored in the TSDB. It is necessary to first transform all the measurements to the same time instant, then define a spatial ROI, and finally generate a message with the data of the vehicles that are inside the ROI. These steps are represented in Algorithm 1.

Algorithm 1 VDS response to a query.

```
{Vehicles} ← queryTSDB()
for each vehicle ∈ Vehicles do
    updatePosition(vehicle)
    if isInsideROI(vehicle) then
        {RelevantVehicles} ← vehicle
    end if
end for
    return {RelevantVehicles}
```

When the VDS receives a request it needs to retrieve from the TSDB the latest measurements stored for each neighbour vehicle in a certain time span. Considering the 10 Hz frequency of CPM messages, the TSDB query should be for at least the latest 100 ms. In Fig. 2 an example of a message timeline of the different actors involved in a VDS query is depicted. $VehicleA$, $VehicleB$ and $VehicleC$ are periodically sending their geo-positioning messages to the VDS. The messages are not synchronised between vehicles, so there is a time offset between for instance t_{Va_m1} and t_{Vb_m1}. The messages contains geo-positioning data and a time stamp to indicate the time instant when this data was measured. $VehicleC$ sends a message to query the VDS. This message contains geo-positioning data of $VehicleC$ captured at t_{q_sent} time instant and is received in the MEC in t_{q_rec}. The latency of the query message can be calculated as:

$$latency_{query} = t_{q\_rec} - t_{q\_sent} \tag{1}$$

In t_{q_rec}, the latest positions stored in the TSDB from $VehicleA$ and $VehicleB$ were captured in t_{Va_m2} and t_{Vb_m1} respectively. In the proposed method, the VDS would retrieve

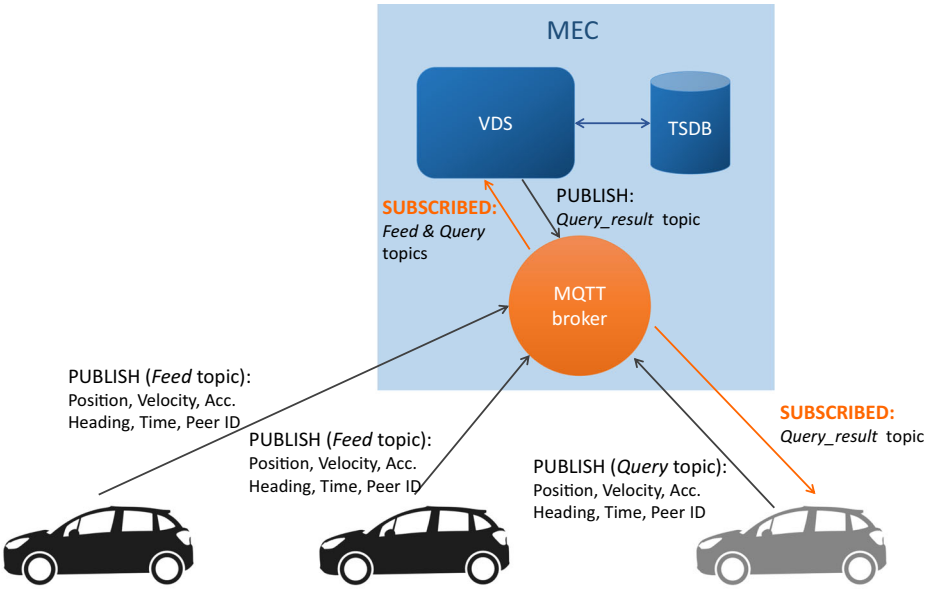


Fig. 1 Architecture of MEC-based vehicle discovery approach

the positions of $VehicleA$ and $VehicleB$ stored in the TSDB and it would transform them to t_{r_rec} time instant, i.e. the time instant when the VDS response reaches $VehicleC$. Then, t_{r_rec} can be calculated as:

$$t_{r\_rec} = t_{q\_sent} + latency_{query} + T_{proc\_query} + latency_{resp} \tag{2}$$

where T_{proc_query} is the time spent by the VDS processing the query, and $latency_{resp}$ is the latency of the response message. t_{q_sent}, $latency_{query}$ and T_{proc_query} are known values for the VDS. $latency_{resp}$ can be considered to be the same than $latency_{query}$, as the response message should still be a lightweight message with the data of only a subset of the connected neighbour vehicles. The position, speed, acceleration, heading and timestamp of $VehicleC$ were included in the query message, so the VDS can estimate he position of $VehicleC$ in

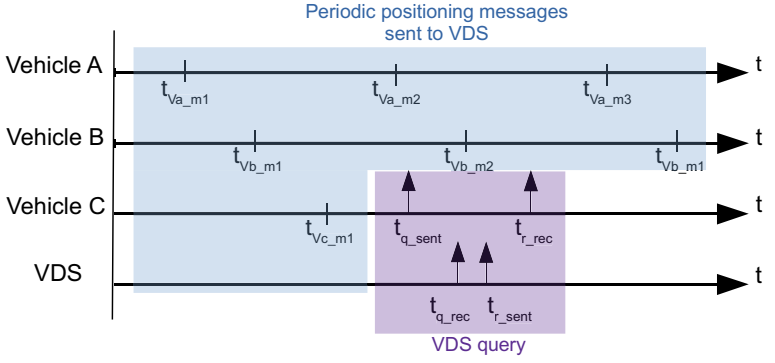


Fig. 2 Example of a message timeline of the different actors involved in a VDS query

t_{r_rec} using the following equations:

$$x_{r\_rec} = x_{q\_sent} + v \cdot cos(\alpha) \cdot (t_{r\_rec} - t_{q\_sent}) + \frac{1}{2} \cdot a \cdot cos(\alpha) \cdot (t_{r\_rec} - t_{q\_sent}) \quad (3)$$

$$y_{r\_rec} = y_{q\_sent} + v \cdot sin(\alpha) \cdot (t_{r\_rec} - t_{q\_sent}) + \frac{1}{2} \cdot a \cdot sin(\alpha) \cdot (t_{r\_rec} - t_{q\_sent}) \quad (4)$$

where (x_{q_sent}, y_{q_sent}) and (x_{r_rec}, y_{r_rec}) are the positions of the vehicle in Cartesian coordinates in t_{q_sent} and t_{r_rec} respectively, and v, a and α are the velocity, acceleration and heading of the vehicle in t_{q_sent}. The positions of the rest of the vehicles stored in the TSDB are also updated to t_{r_rec} using the same Physical equations and calculating the time delta as the time elapsed from the time instant when the vehicle position was measured to t_{r_rec}.

Once all the neighbour vehicle positions are transformed to the same time instant, t_{r_rec}, the VDS checks which vehicles are inside a ROI for the vehicle that requested the service. The ROI defines a spatial region where a vehicle can be located to be considered as interesting for the query response. The present method proposes to use an ellipse to define the ROI because for this application it is more suited to have a lengthened shape rather than a circle or square. In addition, it is fast to compute if a certain point is inside an ellipse. A vehicle positioned at (x, y) can be determined to be inside the ROI if the following equation is satisfied:

$$\frac{((x-h) \cdot cos(\alpha) + (y-k) \cdot sin(\alpha))^2}{m^2} + \frac{((x-h) \cdot sin(\alpha) - (y-k) \cdot cos(\alpha))^2}{n^2} \leq 1 \quad (5)$$

where h, k and a, b are the ellipse shifts and semi-axis in the x and y directions respectively, and α is the heading angle of the vehicle that requested the query measured from x axis. Moreover, m equals to half of the ellipse width and n to half of the ellipse height. Here, the ellipse width corresponds to the maximal longitudinal distance range and the ellipse height to the maximal lateral distance range. These values are defined in the query to the VDS. Example values would be 50 m for the ellipse height, which is aligned with other similar Extended Sensors applications [15], and 14 m for the ellipse height, calculated as four times a standard lane width. The ellipse is oriented in the direction of the heading of the vehicle that queried the VDS, so h and k are calculated as follows:

$$h = cos(\alpha) \cdot m + x_{r\_rec} \quad (6)$$

$$k = sin(\alpha) \cdot m + y_{r\_rec} \quad (7)$$

where (x_{r_rec}, y_{r_rec}) is the position of the vehicle that requested the VDS. The VDS sends in a message the list of neighbour vehicles that are inside the ROI to the vehicle that requested the service. This message includes not only the updated position but also the velocity, acceleration and heading of each vehicle, which are considered to be constant in such small time lapses, and the sensor data source identifier to establish a connection. This information is used by the receiver to start an Extended Sensors service with a specific vehicle and sensor.

4 CAM service implementation

4.1 Data formats & protocol stack

An increasing number of sensors present in vehicles provides information to on-board systems to guide and support the driver for safe driving. Sophisticated ADAS systems ship widely employed sensors such as video cameras, Light Detection and Ranging (LIDAR) and

Global Navigation Satellite System (GNSS). As vehicles become increasingly connected to each other and to external infrastructures, the need to compress and securely transmit produced data streams becomes primary.

The different types of data exchanged by CAM services are:

- Video streams. The set of video cameras shipped by L3/L4 vehicles goes from 4 to 7 to cover different angles and distances. Cameras capture up to 1920x1080 image resolution and up to 60 fps. Some of them only process grayscale images as the colour is not always relevant to detect events or warning situations. Furthermore, this color-space reduction helps to restrain data throughput to be processed.
- Point clouds. LIDAR sensors produce around a million points per second covering a 360^o horizontal field of view and about 30^o vertical field of view (+/- 15^o up and down).
- Geo-position. Advanced positioning systems usually fuse GNSS receivers with other sensors to enhance the accuracy of GNSS and ensure continuity, generating 10-100 Hz samples including latitude/longitude tuple, heading (compass), speed and acceleration.
- CAM service messages. V2X applications exchange data publishing positioning, sensor information, driving dynamics and detected objects using CPM format [12].

For the transmission of previous data, WebRTC streaming protocol is a good option. On top of the stack depicted in Fig. 3, it has been designed to transmit video streams with a data channel [30], which brings several benefits from legacy Real Time Protocol (RTP) and User Datagram Protocol (UDP)-based technologies:

- Network Address Translation (NAT)-transversal. Supported by Session Traversal Utilities for NAT (STUN) and Traversal Using Relay NAT (TURN) servers and managed by Interactive Connectivity Establishment (ICE) standard protocol, data flows can cross private and public networks.
- Secure. Data streams are encrypted using Datagram Transport Layer Security (DTLS) and media streams are encrypted using Secure Real-time Transport Protocol (SRTP), where certificates handshake is end-to-end managed.
- Low latency. It performs real-time delivery on top of RTP/UDP communications.
- Monitored. It includes Stream Control Transport Protocol (SCTP) and Real-time Transport Control Protocol (RTCP) for congestion and flow control providing performance statistics [36].
- Unicast and Multi-party modes. It allows the participation in a unicast communication or to join a multi-party session. This is also important when different vehicles or edge infrastructures wish to concurrently consume a specific sensor data source.

All these features turn WebRTC into an ideal streaming protocol for video streams from cameras, point clouds from LIDAR systems, geo-position data from GPS/GNSS sensors and any CAM service message.

4.2 CAM communications

From the variety of V2X use cases [2], this work focuses on Extended Sensors use case, as it performs the common steps for the initial communication of other use cases (Advanced Driving and Platooning), where the start-up time and the end-to-end latency performance have a big impact.

The Extended Sensors use case that we have implemented is a See-Through application, where a trailing vehicle consume a front video camera from the leading vehicle in order to

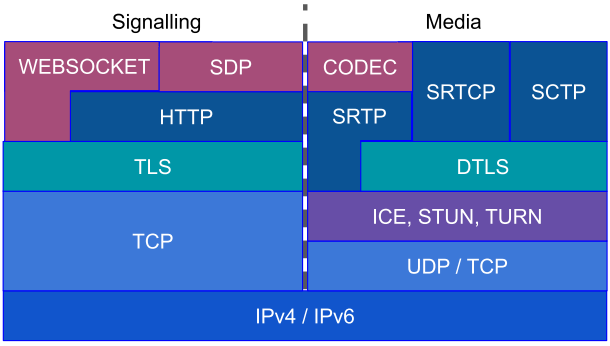


Fig. 3 WebRTC protocol stack for signalling and media

get a better understanding of ahead environment. In this case, to overlay another camera in a consistent way, it is necessary to align different perspectives and fields of view (FOV). To this end, it is required to get the heading and distance from both vehicles in order to consistently overlay the external video stream in the on-board captured video that will be used by ADAS on-board system. The Fig. 4 shows the resulting See-Through composition. Here, $VehicleB$'s front vision overlays $VehicleA$'s front vision, where the offsets and scale depend on the distance and the heading of both vehicles [24]. This way, $VehicleA$ can get awareness of warnings or situations affecting its field of view. Another camera aspects such as aperture, focus distance and resolution are also relevant [22], but the required deformations could be done in real-time easily, without a tangible impact on start-up time or latency.

In order to establish the communication that makes possible to overlay a video stream over the on-board view, it is necessary to discover and select the appropriate sensor data source needed by each vehicle. The Fig. 5 depicts the data flow and the participants involved to establish a CAM communication session between $VehicleA$ and $VehicleB$, where the MEC system facilitates the discovery and presentation from the available ones to start a data stream session.

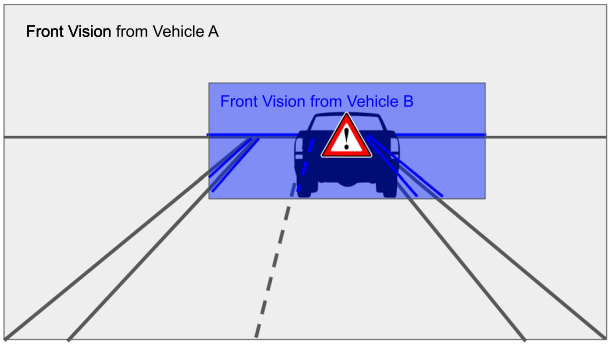


Fig. 4 CAM see-through application

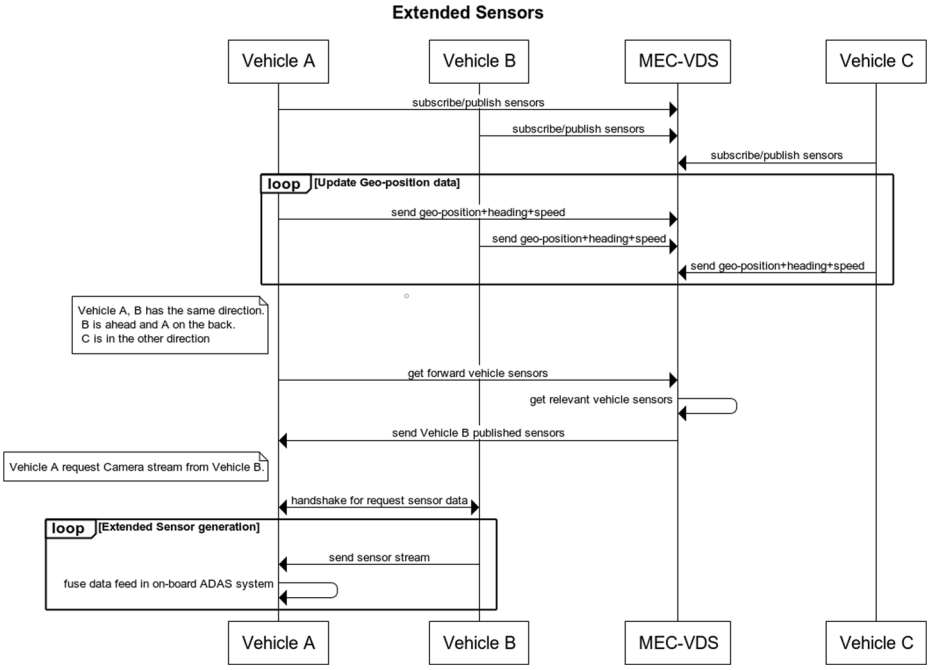


Fig. 5 Example of Extended Sensors message flow

First, vehicles constantly update their geo-position data. Suddenly, $VehicleA$ queries the VDS for relevant sensor data stored at the MEC. Finally, according to the provided information, the sensor stream from $VehicleB$ starts and the received data is used to complement on-board perception of $VehicleA$.

5 Results

5.1 Experimental setup

To assess the advantages of this MEC VDS system in terms of start-up time, end-to-end latency and spatial accuracy, we have deployed a setup to conduct experiments on similar circumstances than a cellular 5G network. Traffic Control [9] is the employed utility to change the uplink and downlink performance to emulate a 5G radio link modifying the bandwidth and the latency of the LAN network interface to 100Mbps and 17ms respectively [16]. So, the experimental setup, depicted in Fig. 6, comprises:

- Vehicle nodes. Automotive-grade equipment (NVIDIA DRIVE PX2) with Ethernet interface features LAN connectivity with limited bandwidth and latency, Ubuntu 18.04 and Gstreamer 1.14 with WebRTC plugins. The system includes a VALEO fish-eye camera capturing 1928x1208 images at 29fps.
- WebRTC Proxy. PC Intel i5-9400F CPU @ 2.90GHz with 8GB RAM and Ubuntu 18.04 running a Python proxy for WebRTC sessions in a testing room.

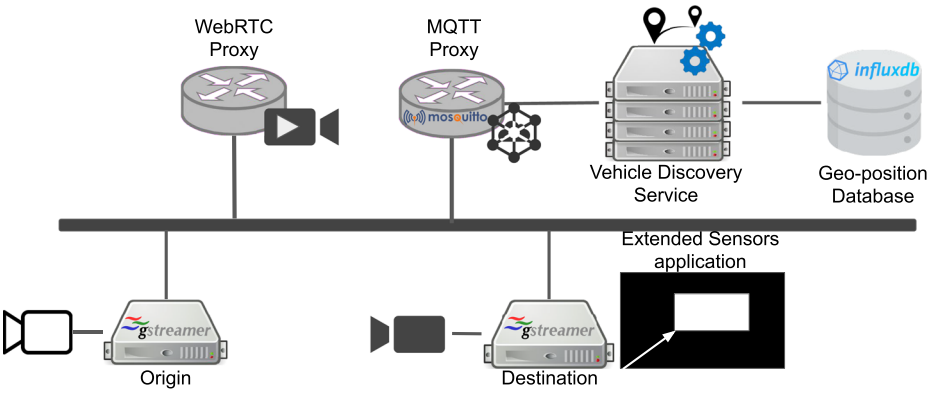


Fig. 6 Experimental setup

– MEC Vehicle Discovery Service. PC Intel i5-7500 CPU @ 3.40GHz with 8GB RAM and Ubuntu 18.04. This service is queried by means of a MQTT server implemented with Eclipse Mosquitto. The VDS exploits a local InfluxDB database which stores the latest geo-positions published to the Mosquitto server.

In order to test the spatial accuracy of the VDS, several scenarios from the CommonRoad dataset [4] have been used. The scenarios are partly recorded from real traffic and partly hand-crafted to create challenging situations. The selected urban scenarios cover three different intersections, and the highway scenarios include lane merges and curves to make them more challenging. More information about the selected CommonRoad dataset scenarios can be found in Table 1.

The dataset provides position, velocity, acceleration and heading values of each vehicle recorded at 10 Hz. These measurements are synchronised between vehicles, meaning that all vehicles provide a measurement exactly at the same time. However, this is not a realistic situation as explained in Section 3. So, a random time offset of less than 100 ms was added to each vehicle.

To test the Extended Sensors service, a video dataset has been used. The dataset consists in outdoors video sequences capturing real urban and highways road traffic using the previously described camera. The total duration of the dataset is 15 minutes and the chosen duration for each test is fixed to 30 seconds, scheduling a live session every 30 seconds, resulting in 30 different session slots.

Table 1 CommonRoad dataset scenarios used to test the spatial accuracy of the VDS

Scenario name	Scenario class	Duration (s)	Number of dynamic vehicles
DEU_Muc-16_1_T-1	Urban	12.1	17
USA_Peach-2_1_T-1	Urban	10.5	16
USA_Peach-4_7_T-1	Urban	12.0	19
CHN_Cho-2_8_T-1	Highway	15.0	26
CHN_Cho-2_10_T-1	Highway	16.5	18
DEU_Stu-1_5_T-1	Highway	16.0	22

5.2 Evaluation metrics

5.2.1 Spatial accuracy metrics

The objective of the VDS is to provide to interested vehicles with a list of relevant vehicles inside a region of interest that are ready to establish a Extended Sensors session. As explained in Section 3, due to vehicle dynamics, a delay on the position data can mean a significant displacement in the spatial domain. If the network latencies and processing times are not properly considered, the list of vehicles provided by the VDS is outdated by the time it reaches the receiver. Moreover, positioning errors in the order of metres can make a vehicle to be located in a wrong lane or road. This could imply significant errors in the vehicle list provided in the VDS response: some vehicles should not be there (false positives) or some vehicles are missing (false negatives).

Three options for VDS implementation have been compared:

- Option 1: the VDS just forwards the latest value stored at the TSDB for each vehicle. It does not consider the time shifts between vehicles.
- Option 2: the VDS transforms the latest values stored at the TSDB to the time instant when the query is received in the MEC. Following the terminology defined in Section 3.1, this time instant would be t_{q_rec}.
- Option 3 (method proposed in Section 3): the VDS transforms the latest values stored at the TSDB to the time instant when the query response is estimated to be received in the vehicle. Following the terminology defined in Section 3.1, this time instant would be t_{r_rec}.

Each of these options is assessed using a LTE network latency (38 ms) [28] and a 5G network latency (17 ms) [16]. A further discussion about network latencies is included in Section 5.4. For each combination of VDS implementation and network latency, the Euclidean distance between each vehicle position sent by the VDS and the actual vehicle position at the time that the VDS message is received (t_{r_rec}) is calculated. This Euclidean distance represents the position error. The distribution of position errors obtained per vehicle and time interval can be analised using statistical metrics. Here we propose the most common ones: median, mean, standard deviation (SD) and maximum.

5.2.2 Start-up & latency metrics

In order to provide a useful data flow to complement the on-board perception in time, it is important to minimize start-up time and to keep external data stream delay under the time for one sample to behave as a local sensor.

For the start-up time, the time consumption by MQTT queries to discover the required sensor data sources and Peer IDs, and the WebRTC session establishment are the two main factors to take into account. According to the protocol described in Fig. 5, the messages exchanged are minimized to alleviate the discovery messaging, then the standard WebRTC message protocol is triggered. This protocol includes several steps for a *call* [37]:

$$\begin{aligned} startup_{webrtc} = f(&t^{orig}_{offer,stun}, t^{orig\rightarrow dest}_{send,tcp}, t^{dest}_{accept}, t^{dest}_{answer,stun}, t^{dest\rightarrow orig}_{send,tcp}, t^{orig}_{accept}, \\ &t^{orig\leftrightarrow dest}_{endpoints,ice}, t^{orig\leftrightarrow dest}_{handshake,dtls}, t^{orig\rightarrow dest}_{datachannel,sctp}, t^{orig\rightarrow dest}_{video,srtp}) \end{aligned} \tag{8}$$

Where each parameter means:

1. Create Offer ($t^{orig}_{offer,stun}$). The origin allocates IP address and ports with STUN/TURN and accordingly generates the manifest Session Description Protocol (SDP) with all the streaming information included.
2. Send Offer ($t^{orig \rightarrow dest}_{send,tcp}$). The origin sends the *Offer*, including the SDP, to the destination by a TCP connection.
3. Process Offer (t^{dest}_{accept}). The destination accepts the *Offer* and parses the SDP.
4. Create Answer ($t^{dest}_{answer,stun}$). The destination creates an *Answer*. To this end, the destination allocates IP address and ports with STUN/TURN and accordingly generates the manifest SDP.
5. Send Answer ($t^{dest \rightarrow orig}_{send,tcp}$). The destination sends the *Answer*, including the SDP, to the origin by a TCP connection.
6. Process Answer (t^{orig}_{accept}). The origin accepts the *Answer* and parses the SDP.
7. Endpoints check ($t^{orig \leftrightarrow dest}_{endpoints,ice}$). Both sides perform ICE checks to establish UDP connections according to the SDP to start the SRTP streams.
8. Credentials handshake ($t^{orig \leftrightarrow dest}_{handshake,dtls}$). DTLS protocol is performed and credentials for SRTP communications are exchanged.
9. Data Channel ($t^{orig \rightarrow dest}_{datachannel,sctp}$). SCTP communication channel is established for the TCP data channel.
10. Streams run ($t^{orig \rightarrow dest}_{video,srtp}$). SRTP communications deliver the video stream over UDP connections.

As it can be concluded the WebRTC data-flow is much more complex and sophisticated than the regular Real-Time Session Protocol (RTSP) one, widely used by Closed-circuit television (CCTV) cameras. However, it brings essential features listed in Section 4.1.

Concerning the latency, the Fig. 7 shows the different scenarios that would apply. In the left one, ADAS system of *VehicleA* extends its perception with an external data source from *VehicleB* with a timing as it would be on-board. Here, sample rate at *VehicleA* provides enough time to capture data at vehicle B and delivery, before data are needed to be fused at *VehicleA*. In the middle, a likely situation means data is captured at *VehicleB* so closed in a common timeline to the next sample captured at *VehicleA*, that it is not possible to deliver it in time. So, this data can be fused with the next sample captured at *VehicleA* without bringing significant inconsistencies. In the right, a scenario where latency can produce inconsistencies with deprecated external data is shown.

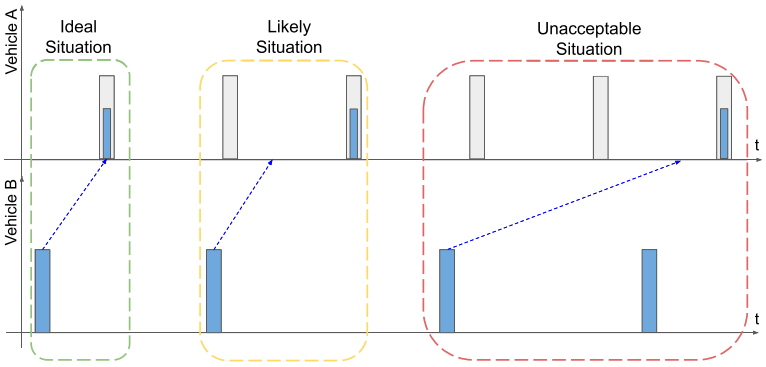


Fig. 7 Latency performance scenarios impact on Extended Sensors application

The end-to-end (e2e) latency is declared as the time elapsed from a instant the data were captured at origin to the moment they are present at destination to be used and interpreted [8]:

$$latency_{e2e} = t^{dest}_{presentation} - t^{orig}_{capturing} \tag{9}$$

Where the assumption to avoid CAM application inconsistencies is that the time elapsed from the moment the data is captured by a sensor at *VehicleB* to the moment they are received and used by a system at *VehicleA*, is limited by the sample rate of *VehicleA* (Fig. 7 right):

$$latency_{e2e} < 2 * \frac{1}{samplerate_{dest}} \tag{10}$$

The (11) shows the main factors that come into place, specifically for video streams:

$$latency_{e2e} = t^{orig}_{enc} + t_{network} + t^{dest}_{buffer} + t^{dest}_{dec} + t^{dest}_{composition} \tag{11}$$

So the different timestamps which are involved are:

- Video capturing time ($t^{orig}_{capturing}$). In order to have a consistent timeline between participants video pipelines are synchronised to Network Time Protocol (NTP). This timestamp is captured, rendered in the image and registered.
- Video encoding time (t^{orig}_{enc}). This includes the time for compression. In our case, H.264 video encoding is employed because it is widely supported by GPU graphics cards for accelerated encoding. In order to minimize latency no reordered frames are used, limited to keyframes (I) and temporal prediction (P) frames. This time is reported by performance metrics from the encoder.
- Network delivery time ($t_{network}$). This is mainly related to the network latency and jitter. The RTCP reports provides different network stats related to Round Trip Time (RTT), beyond the end-to-end NTP-based time shift which is measured by the CAM application at destination. It can be assumed that the RTT is 2 times the network latency, as the latency in both directions are fairly equivalent.
- RTP jitter buffering (t^{dest}_{buffer}). In order to prevent from UDP packets reorder or drops, RTP based communications usually apply a buffer which accommodates network jitter. In our case this is declared under the duration of a video frame. The employed 10 ms helps to enforce communications reliability while keeping impact on end-to-end latency low.
- Video decoding time (t^{dest}_{dec}). This includes the time for decompression. Again, H.264 video decoding is widely supported by GPU graphics cards for accelerated decoding. This time is provided by performance metrics from the decoder.
- Video stitching ($t^{dest}_{composition}$). This process fuses the external data source with the captured on-board data flow and produces a cooperative perception extending on-board sensors.

5.3 Spatial accuracy results

These steps were followed to obtain the results presented in this section:

- Feed the TSDB with a dataset.
- Query the TSDB in 100ms time intervals during the duration of the dataset.
- For each query, apply the three VDS implementations described in Section 5.2.1 with the two network latency options. Thus, six combinations of VDS implementations and network latency. For the third VDS implementation, a processing time of 18 ms was considered, calculated with the hardware described in Section 5.1.

– Calculate the Euclidean distance between each vehicle position included in the VDS response and the actual position at the time that the message reaches the receiver. The actual position or ground truth value is calculated using a quadratic interpolation between the closest points that are part of the dataset.

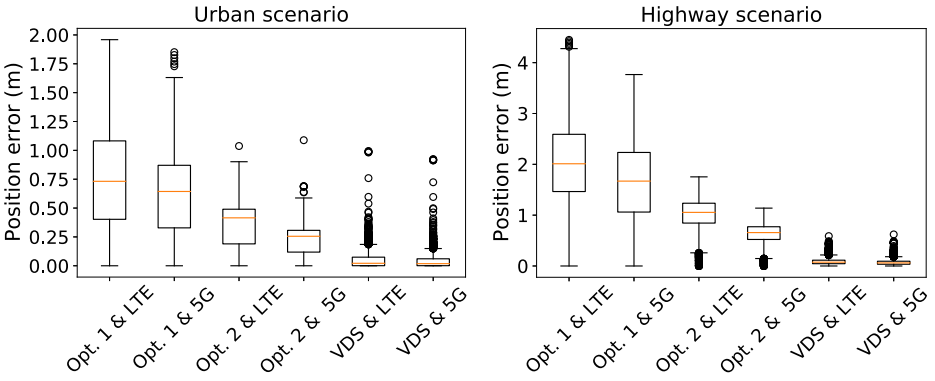
The results are summarised in Table 2. The results obtained by each scenario class, urban or highway, have been aggregated. It is noticeable that using the same network configuration, the proposed VDS method obtain much better results than the other two options. The proposed method obtains sub-metre accuracy in both scenario classes, even with the LTE network latency. This sub-metre accuracy meets the needs of the present vehicle discovery application. It is important to consider that VDS offers a list of geo-positioned vehicles, but these geo-positions will be updated periodically with no intervention of the VDS once the Extended Sensors service is started. However, for the sake of assessing the possibilities of the VDS for more demanding use cases, the obtained accuracy results can be compared with the more stringent localisation requirements for safety-critical autonomous driving manoeuvres. According to [32], an error bound of 0.29 m is required in urban environments, and a lateral error bound of 0.57 m and a longitudinal bound of 1.40 m in highway environments. As it can be noted looking at Fig. 8, the urban error bound is fulfilled by the proposed method in the conducted tests except for some outliers. Regarding the highway environment, the limits proposed by [32] mean an maximal Euclidean error of 1.06 m, which is also satisfied by the proposed VDS.

5.4 Start-up & latency results

First, Table 3 shows the total start-up time and the elapsed time for the individual steps measured from logs of GStreamer peers under 3 possible latency performance. Here, we can see how the network latency performance impacts on the different times, from 5G e2e latency value of 17 ms coming from field tests of industrial setups [16], to LTE e2e latency performance (38 ms) coming from the performance on LTE networks [28].

Table 2 Accuracy results of provided geo-positions for the three methods tested by the VDS service

VDS query response method	Scenario	Network latency	4Median error (m)	Mean error (m)	SD error (m)	Max. error (m)
Forward TSDB values (Opt. 1)	Urban	LTE (38ms)	0.732	0.721	0.468	1.959
Forward TSDB values (Opt. 1)	Urban	5G (17ms)	0.644	0.595	0.384	1.851
Transf. values to t_{q_rec} (Opt. 2)	Urban	LTE (38ms)	0.415	0.352	0.211	1.038
Transf. values to t_{q_rec} (Opt. 2)	Urban	5G (17ms)	0.256	0.221	0.135	1.088
Transf. values to t_{r_rec} (VDS)	Urban	LTE (38ms)	0.021	0.050	0.074	0.994
Transf. values to t_{r_rec} (VDS)	Urban	5G (17ms)	0.018	0.043	0.067	0.924
Forward TSDB values (Opt. 1)	Highway	LTE (38ms)	2.011	2.029	0.788	4.448
Forward TSDB values (Opt. 1)	Highway	5G (17ms)	1.670	1.668	0.743	3.766
Transf. values to t_{q_rec} (Opt. 2)	Highway	LTE (38ms)	1.055	1.009	0.303	1.753
Transf. values to t_{q_rec} (Opt. 2)	Highway	5G (17ms)	0.657	0.629	0.192	1.138
Transf. values to t_{r_rec} (VDS)	Highway	LTE (38ms)	0.075	0.086	0.061	0.621
Transf. values to t_{r_rec} (VDS)	Highway	5G (17ms)	0.061	0.070	0.052	0.587



(a) Urban scenario (b) Highway scenario

Fig. 8 Accuracy results of provided geo-positions for the three methods tested by the VDS service

It is clear that the radio link latency brings a significant impact as it affects to several sequenced messages exchanged between peers and also sent to the servers. It is also evident that the STUN and the ICE protocol to find the connection endpoints crossing the infrastructures is a heavy mechanism meaning high times. Then, the DTLS handshake is also time consuming including most of the messages to establish the SCTP-based data channel. As some steps overlap in time, depending on the implementation, the total time is lower than the aggregation of individual steps.

Concerning the e2e latency for video, Table 4 shows the different time consumption for the identified video transmission steps.

The $latency_{e2e}$ shows the drift between the sent NTP timestamp of a captured frame in the origin and the NTP timestamp when this frame is used at destination. The average value is 62ms and the standard deviation is 8.2 ms. So, the average value is under the 69 ms threshold (for 29 fps) that would ensure a maximum shift of one frame. However, this is not satisfied by the 0,01% of frames.

Table 3 Average measured Start-up time and partial times for components of (8)

Start-up step	time for wired (no added latency)	time for 5G latency (17ms)	time for LTE latency (38ms)
$t^{orig}_{offer,stun}$	385ms	316ms	387ms
$t^{orig \rightarrow dest}_{send,tcp}$	60ms	78ms	99ms
t^{dest}_{accept}	4ms	4ms	4ms
$t^{dest}_{answer,stun}$	52ms	107ms	173ms
$t^{dest \rightarrow orig}_{send,tcp}$	56ms	78ms	108ms
t^{orig}_{accept}	1ms	3ms	3ms
$t^{orig \leftrightarrow dest}_{endpoints,ice}$	257ms	376ms	516ms
$t^{orig \leftrightarrow dest}_{handshake,dtls}$	150ms	208ms	258ms
$t^{orig \rightarrow dest}_{datachannel,sctp}$	4ms	18ms	13ms
$t^{orig \rightarrow dest}_{video,srtp}$	50ms	67ms	88ms
$startup_{webrtc}$	658ms	737ms	948ms

Table 4 Average measured pipeline times

Pipeline step	time for 5G latency (17ms)
$t^{orig}_{capturing}$	1ms
t^{orig}_{enc}	30ms
$t_{network}$	17ms
t^{dest}_{buffer}	10ms
t^{dest}_{dec}	3ms
$t^{dest}_{composition}$	1ms
$latency_{e2e}$	62ms

6 Discussion

A MEC-centred approach can efficiently collect and store the positioning data fed by the surrounding vehicles to offer a vehicle discovery service to them. In order to select the optimal candidates to establish a CAM service session between vehicles, it is important to consider the timestamps of the vehicle data stored at the MEC database and the latencies of the communication pipeline. If none of this is considered, significant errors in the order of metres can happen and these errors can led to the selection of wrong candidates. This is specially relevant fir the highway scenario, where higher velocities imply higher displacements.

If the service deployed at the MEC transforms the vehicle geo-positions to the time instant when the query is received, the accuracy is improved. However, the deviation with the actual vehicle position in the instant the response message is received is still significant, especially in the highway scenario with the LTE network latency.

The method proposed in this paper for vehicle discovery obtains very good results both with LTE and 5G network latencies. In urban scenarios, a mean error of 5 cm (SD = 7.4 cm) with LTE and 4.3 cm (SD = 6.7 cm) with 5G is obtained when comparing the query responses with the actual vehicle positions. In highway scenarios, a mean error of 8.6 cm (SD = 6.1 cm) with LTE and 7 cm (SD = 5.2 cm) with 5G is obtained. The positioning accuracy bounds defined by the literature for the safety-critical autonomous driving applications are fully satisfied in the highway scenario (error $<$ 1.06 m) in all the samples tested in the experiments. In the urban scenario, the error bound (error $<$ 0.29 m) is satisfied for all the samples except for some outliers. The reason of this is that the dynamics in urban scenarios are more complex to predict with sharp turns and frequent changes in the acceleration and heading. In order to improve the accuracy, more precise methods for trajectory prediction can be implemented like a Kalman Filter or a Long Short Term Memory (LSTM) neural network. However, these methods require significantly higher computation loads and could be challenging to implement considering that the MEC could potentially serve several vehicles at the same time. A trade-off between accuracy and computational load is necessary. The proposed VDS is quick (18 ms of processing time) and accurate enough for the See-Through application presented in this study.

The advantage of using 5G over LTE is more evident looking at the start-up times of the implemented See-Through application. This time is reduced from 948 ms to 737 ms, which is a significant improvement. These start-up times may seem too high, but they are enough for starting an Extended Sensors application. Moreover, the features provided by WebRTC compensate the start-time overheads that it brings.

Finally, the proposed WebRTC-based streaming technology satisfies latency constraints for the tested Extended Sensors application. However, the current benchmarks of industry deployments supply a tight performance violating in some frames the frame-level latency requirements. So, further improvements in 5G Radio deployments providing the promised performance of 1ms latency would bring enforced reliability, essential for this kind of applications. Moreover, as the latency and jitter from the radio links get lower, the buffering time to prevent from jitter issues and to accommodate packets at network interface could be reduced.

7 Conclusion

The MEC-based lightweight vehicle discovery service presented in this paper is able to perform well even with LTE network latencies. In addition to centralising the vehicle queries, the MEC also provides benefits with respect to data privacy and security as the data flows are restricted to a geographical area close to the user. As demonstrated in this paper, the stringent video streaming requirements of a Extended Sensors use case need beyond-LTE capabilities that can be fulfilled with 5G.

The MEC services might become key actors to negotiate QoS communications between cars not only based on radio availability but considering hardware capacity on destination. However, some points still need more research and standardisation effort. For instance, there are still no standard mechanisms to synchronise data between MECs, which is a key aspect with mobile nodes such as vehicles.

Acknowledgements This work is a part of the 5G-MOBIX project. This project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825496. Content reflects only the authors' view and European Commission is not responsible for any use that may be made of the information it contains.

References

1. 3GPP (2018) 3GPPP TS 22.886 version 16.2.0 - Study on enhancement of 3GPP support for 5G V2X services. http://www.3gpp.org/ftp//Specs/archive/22_series/22.886/22886-g20.zip. Accessed 25 May 2020
2. 3GPP (2019) 3GPPP TS 22.186 version 16.2.0 - Enhancement of 3GPP support for V2X scenarios. http://www.3gpp.org/ftp//Specs/archive/22_series/22.186/22186-g20.zip. Accessed 25 May 2020
3. Abbas F, Fan P, Khan Z (2019) A novel Low-Latency V2V resource allocation scheme based on cellular V2X communications. IEEE Trans Intell Transport Syst 20(6):2185–2197. https://doi.org/10.1109/TITS.2018.2865173
4. Althoff M, Koschi M, Manzinger S (2017) Commonroad: Composable Benchmarks for Motion Planning on Roads. Proc. of the IEEE Intelligent Vehicles Symposium. 719–726. https://doi.org/10.1109/IVS.2017.7995802
5. Arregui H, Mujika A, Loyo E, Velez G, Barros MT, Otaegui O (2019) Short-term vehicle traffic prediction for Terahertz Line-of-Sight estimation and optimization in small cells. IEEE Access 7:144408–144424. https://doi.org/10.1109/ACCESS.2019.2910225
6. Barros MT, et al. (2020) CogITS: cognition-enabled network management for 5G V2X communication. IET Intell Transp Syst 14(3):182–189. https://doi.org/10.1049/iet-its.2019.0111
7. Boban M, Kousaridas A, Manolakis K, Eichinger J, Xu W (2018) Connected roads of the future: use cases, requirements, and design considerations for vehicle-to-everything communications. IEEE Veh Technol Mag 13(3):110–123. https://doi.org/10.1109/MVT.2017.2777259

8. Bulfone A, Drioli C, Ferrin G, et al. (2020) A scalable system for the monitoring of video transmission components in delay-sensitive networked applications. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-020-08743-7
9. Canonical Ubuntu (2019) Traffic Control. http://manpages.ubuntu.com/manpages/bionic/man8/tc.8.html. Accessed 25 May 2020
10. ETSI (2016) ETSI GS MEC 002 version 1.1.1 - Mobile Edge Computing (MEC): Technical Requirements. https://www.etsi.org/deliver/etsi_gs/MEC/001_099/002/01.01.01_60/gs_MEC002v010101p.pdf. Accessed 25 May 2020
11. ETSI (2017) 3GPP TS 22.185 version 14.3.0 Release 14 - Service requirements for V2X services. https://www.etsi.org/deliver/etsi_ts/122100_122199/122185/14.03.00_60/ts_122185v140300p.pdf. Accessed 25 May 2020
12. ETSI (2019) ETSI TR 103 562 version 2.1.1 - Vehicular Communications, Basic Set of Applications: Analysis of the Collective Perception Service (CPS) Release 2. https://www.etsi.org/deliver/etsi_tr/103500_103599/103562/02.01.01_60/tr_103562v020101p.pdf. Accessed 25 May 2020
13. Fodor G, et al. (2019) Supporting enhanced vehicle-to-everything services by LTE release 15 systems. IEEE Communications Standards Magazine 3(1):26–33. https://doi.org/10.1109/MCOMSTD.2019.1800049
14. Ghosh A, Maeder A, Baker M, Chandramouli D (2019) 5G Evolution: A View on 5G Cellular Technology Beyond 3GPP Release 15. IEEE Access 7:127639–127651. https://doi.org/10.1109/ACCESS.2019.2939938
15. Gomes P, Vieira F, Ferreira M (2012) The See-Through System: From implementation to test-drive. IEEE Vehicular Networking Conference (VNC). Seoul, 40–47. https://doi.org/10.1109/VNC.2012.6407443
16. GSM Alliance (2017) Cellular Vehicle-to-Everything (C-V2X) enabling intelligent transport. https://www.gsma.com/iot/wp-content/uploads/2017/12/C-2VX-Enabling-Intelligent-Transport_2.pdf. Accessed 25 May 2020
17. Higuchi T, Giordani M, Zanella A, Zorzi M, Altintas O (2019) Value-Anticipating V2V Communications for Cooperative Perception. 2019 IEEE Intelligent Vehicles Symposium (IV):1947–1952. https://doi.org/10.1109/IVS.2019.8814110
18. IETF (2020) IPWAVE WG Vehicular Neighbor Discovery for IP-Based Vehicular Networks. https://tools.ietf.org/html/draft-jeong-ipwave-vehicular-neighbor-discovery-10. Accessed 25 Jan 2021
19. Kim D, Kim S (2020) Incoming Traffic Control of Fronthaul in 5G Mobile Network for Massive Multimedia Services. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-020-08793-x
20. Klaine PV, Imran MA, Onireti O, Souza RD (2017) A survey of machine learning techniques applied to Self-Organizing cellular networks. IEEE Commun Surv Tutor 19(4):2392–2431. https://doi.org/10.1109/COMST.2017.2727878
21. Lee YS, Kim S (2019) A proposal of novel design on the WAVE MAC algorithm with low-latency for seamless video surveillance in V2X environment. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-019-7151-1
22. Li D, Song D, Liu S, et al. (2020) Camera pose estimation based on global structure from motion. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-020-09045-8
23. Light RA (2017) Mosquitto: server and client implementation of the MQTT protocol. The Journal of Open Source Software 2(13). https://doi.org/10.21105/joss.00265
24. Lin L, Ding Y, Wang L, et al. (2019) Line-preserving video stitching for asymmetric cameras. Multimed Tools Appl 78:14591–14611. https://doi.org/10.1007/s11042-018-6848-x
25. Liu C, Zhang G, Guo W, He R (2020) Kalman Prediction-Based neighbor discovery and its effect on routing protocol in vehicular ad hoc networks. IEEE Trans Intell Transport Syst 21(1):159–169. https://doi.org/10.1109/TITS.2018.2889923
26. Martin A, et al. (2018) Network Resource Allocation System for QoE-Aware Delivery of Media Services in 5G Networks. IEEE Trans Broadcast 64(2):561–574. https://doi.org/10.1109/TBC.2018.2828608
27. Martin A, Viola R, Zorrilla M, Flórez J, Angueira P, Montalbán J (2019) MEC For fair, reliable and efficient media streaming in mobile networks. IEEE Trans Broadcast 66(2):264–278. https://doi.org/10.1109/TBC.2019.2954097
28. Martín-Sacristán D et al (2020) Low-Latency Infrastructure-Based Cellular V2V Communications for Multi-Operator Environments With Regional Split. IEEE Transactions on Intelligent Transportation Systems. https://doi.org/10.1109/TITS.2019.2962097
29. McKinsey & Company (2018) From buzz to bucks – automotive players on the highway to car data monetization. McKinsey Center for Future Mobility. https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/accelerating-the-car-data-monetization-journey. Accessed 25 May 2020

30. Palacios S, Santos V, Barsallo E, et al. (2019) Miostream: a peer-to-peer distributed live media streaming on the edge. Multimed Tools Appl 78:24657–24680. https://doi.org/10.1007/s11042-018-6940-2
31. Porambage P, Okwuibe J, Liyanage M, Ylianttila M, Taleb T (2018) Survey on Multi-Access edge computing for internet of things realization. IEEE Commun Surveys Tutor 20(4):2961–2991. https://doi.org/10.1109/COMST.2018.2849509
32. Reid TG, Houts SE, Cammarata R, Mills G, Agarwal S, Vora A, Pandey G (2019) Localization Requirements for Autonomous Vehicles. SAE International Journal of Connected and Automated Vehicles 2(3):173–190. https://doi.org/10.4271/12-02-03-0012
33. Thandavarayan G, Sepulcre M, Gozalvez J (2019) Analysis of Message Generation Rules for Collective Perception in Connected and Automated Driving. 2019 IEEE Intelligent Vehicles Symposium (IV):134–139. https://doi.org/10.1109/IVS.2019.8813806
34. Thompson RL, Hu Z, Cho J, Stovall J et al (2018) Enhancing Driver Awareness Using See-Through Technology. SAE Technical Paper 2018-01-0611. https://doi.org/10.4271/2018-01-0611
35. Velez G, Martin Á, Pastor G, Mutafungwa E (2020) 5G Beyond 3GPP Release 15 for Connected Automated Mobility in Cross-Border Contexts. Sensors 20(22):6622. https://doi.org/10.3390/s20226622
36. Vivekananda GN, Chenna-Reddy P, Ilknur A (2019) A congestion avoidance mechanism in multimedia transmission over MANET using SCTP multi-streaming. Multimedia Tools and Applications. https://doi.org/10.1007/s11042-019-7260-x
37. W3C (2019) WebRTC 1.0: Real-time Communication Between Browsers. https://www.w3.org/TR/webrtc/. Accessed 25 May 2020
38. Yan Z, Li H, Nakazato H, Park Y, Lee J (2019) DNS based Neighbor Discovery in ITS. 2019 IEEE International Conference on Consumer Electronics (ICCE):1–2. https://doi.org/10.1109/ICCE.2019.8662016