Enabling Computation Offloading for Autonomous and Assisted Driving in 5G Networks

Estefanía Coronado*, Gabriel Cebrián-Márquez‡, and Roberto Riggio*
*Wireless and Networked Systems, Fondazione Bruno Kessler, Trento, Italy
Email: {e.coronado, rriggio}@fbk.eu
‡Department of Computer Science. University of Oviedo, Oviedo, Spain
Email: cebriangabriel@uniovi.es

***Abstract*—Connected and automated vehicles currently leverage on-board resources to implement autonomous and assisted driving operations. Such functionalities, which are characterized by tight latency demands, require significant processing resources and can generate a considerable amount of data. Cloud computing is considered the one-stop solution for executing computationally intensive workloads. However, accommodating autonomous and assisted driving requirements using a centralized cloud computing platform is not always feasible due to the latency and reliability constraints they impose. In this paper, we introduce a multi-access edge computing platform suitable for offloading certain autonomous and assisted driving tasks to the edges of the network. We also illustrate how both paradigms (centralized and edge cloud computing) can coexist complementing each other in the challenging task of supporting autonomous and assisted driving, thus opening up new horizons for connected vehicles, for which service instantiation and migration needs to be seamless due to its impact on road safety.**

***Index Terms*—5G, MEC, Cloud Computing, Autonomous and Assisted Driving, Computer Vision.**

I. Introduction

The evolution of the automobile industry is heralding the dawn of one of the most relevant transformations in our society in the search for higher safety, efficiency and user experience. While self-driving vehicles seemed a science-fiction scenario some years ago, they have currently become a tangible reality. Despite their driverless capabilities, these *autonomous* vehicles have introduced the need to interchange information to cooperate and ensure driving safety in what is known as Cooperative, Connected, and Automated Mobility (CCAM).

Vehicle to Vehicle (V2V), Vehicle to Infrastructure (V2I) and Vehicle to Everything (V2X) are just a few examples of models that enable vehicular intercommunication. While ITS-G5 is currently the main option for vehicular connectivity, its short range and low performance makes it unsuitable for many applications [1]. By contrast, a variety of advanced features make 5G the key to success of the smart mobility ecosystem. A prime example is the support for Ultra-Reliable Low-Latency Communication (URLLC), a crucial condition for mission-critical applications with stringent performance and reliability requirements, as is the case of CCAM applications. As a matter of fact, specifications for Cellular-based V2X (C-V2X) have been already included in Release 14 [2].

Autonomous vehicles generate enormous amounts of data, e.g., sensor data and video streams to recognize other vehicles, road conditions and external elements such as lanes, and pedestrians [3]. Many other use cases include self-contained operations such as emergency braking and remote driving, and cooperative operations such as lane merging or assisted overtaking, where it is vital to process data from a global point of view. In this context, computer vision has become an irreplaceable technology [4]. At present, however, only high-end vehicles can cope with the significant computing power required using on-board resources, relegating the network to infotainment applications.

Cloud computing has been envisioned by automotive and communication industries as an option to offload storage and compute-intensive applications for battery-constrained devices. However, accommodating automotive services in the cloud may make latency an unbearable challenge. This is rather mitigated by Multi-access Edge Computing (MEC) by offloading such tasks to the edges. Nevertheless, given the limited resources, both paradigms may coexist and complement each other, opening up new horizons for connected vehicles, for which service instantiation and migration needs to be seamless to ensure road safety. This computing combination has been analytically studied in the past [5], [6], but the latency constraints imposed by these services are usually overlooked.

To delve with these limitations, the contribution of the work is three-fold. First, we present a Software-Defining Networking (SDN) architecture for 5G-enabled vehicular networks compliant with the European Telecommunications Standards Institute (ETSI) MEC model. This design allows seamless application instantiation on either the network edge or a remote cloud on the basis of low-latency and performance requirements. Second, we implement a computer vision application for autonomous driving enabling capabilities on lane line detection and on-road object recognition. Finally, we perform an experimental evaluation showing the significant improvements of combining MEC and cloud computation offloading in the context of connected vehicles.

The rest of the paper is outlined as follows. Section II discusses the related work. Section III describes the platform and the offloading model proposed for 5G vehicular networks. The predictive model for autonomous driving is presented in Sec. IV. Section V analyzes the system performance for various offloading models. Finally, Sec. VI draws the conclusions.

II. Related Work

In recent years, computation outsourcing has been drawing increasing attention to reduce data processing time. This has been proved particularly necessary for resource-limited mobile devices and networks, where it is difficult to cope with computationally demanding applications like Augmented Reality (AR) and automation [7].

In the case of vehicular networks, different solutions on the integration of connected vehicles within the cloud-assisted paradigm have been proposed [8]. In this respect, in [9] the data rate of a car-to-cloud communication model is evaluated via simulation, where vehicles transfer sensor data to a remote cloud. However, it does not perform any autonomous driving operation beyond the data analysis. Similarly, [10] introduces a pricing-based matching algorithm by leveraging fog computing for computation offloading in low-latency massive vehicular connectivity. Deeper attention to the automotive requirements in 5G networks is paid in [11], which relies on SDN and OpenFlow features in order to associate vehicles to distributed cloud servers based on mobility patterns, experienced quality and speed.

Despite the benefits of backbone offloading, the main concern resides on the overhead induced. In this respect, MEC is gaining increasing reputation to solve this problem by placing resources closer to the vehicles. This fact is reflected in a huge body of research [12], [13], [14], and standardization from entities such as ETSI [15] and the 5G Automotive Association (5GAA) [16]. In the specific scenario of V2X communications in 5G networks, this approach has been already discussed in detail, as it is the case of [17], which presents several MEC models shared among telecommunications and automotive stakeholders, and outlines the technical requirements in both access and core network.

Given the importance of ensuring ultra-low latency and fast response, partial offloading models considering local, MEC or cloud resources have been also explored. This requires advanced features in the mobile network control plane in order to accommodate changes across computing resources transparently, and an agnostic architecture capable of disposing the traffic from the data plane regardless of such changes [14]. In this respect, [6] introduces a scheme of compressed offloading combining local and MEC computing on the basis of available resources. A similar idea is analytically presented in [5], which also considers neighboring vehicles for task offloading. Nevertheless, most of the current works are still in an early stage, or have been validated via numerical analysis or simulation, making it difficult to evaluate on real deployments.

In order to make CCAM a reality, communication technologies need additional enablers to process the huge amounts of data generated from the road and the vehicles. In this respect, computer vision techniques, and mathematical and Machine Learning (ML) models, are mostly used for this task. However, research on on-road recognition usually overlooks the existing complexities in the communication between vehicles and network infrastructure [18], [19], [20]. Moreover, most of the works are simulation-based, therefore not considering their applicability to real environments.

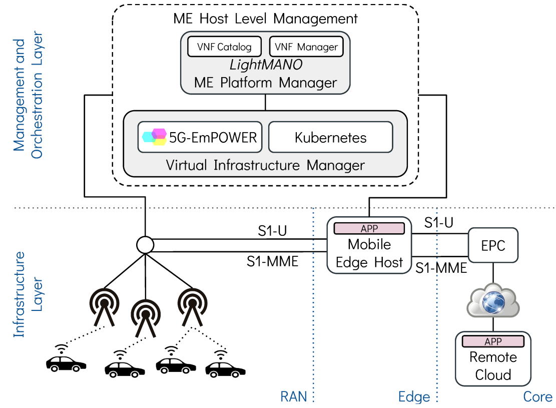


Fig. 1. High-level network view of the system architecture.

This work, in comparison to the aforementioned approaches, puts together a 5G network architecture with ETSI compliance capable of offloading compute workloads to the edge or the cloud, with an experimental computer vision application that is able to fulfill a significant set of relevant use cases in the autonomous driving scenario. Moreover, the effectiveness of the framework is experimentally evaluated and validated.

III. Software-Defined Platform for 5G Networks

A. System Architecture

The system architecture introduced in this work, and shown in Fig. 1, presents two clear dimensions. Depending on the elements' location, Radio Access Network (RAN), network edge, and core network are differentiated, while regarding the functionality, two layers are distinguished, namely infrastructure, and management and orchestration layers.

Infrastructure Layer. Comprises the radio access nodes, which are connected to an Ettus Research Universal Software Radio Peripheral (USRP) b210 using srsLTE, since no open-source 5G stacks are currently available. Radio nodes are connected to a Mobile Edge (ME) Host following a bump-in-the-wire approach [21]. The Evolved Packet Core (EPC) is implemented using nextEPC, and connected to a remote cloud. Complex computation tasks can be offloaded to both the ME Host and the cloud.

Management and Orchestration Layer. Covers the functions of the MEC Platform Manager and the Virtual Infrastructure Manager (VIM). LightMANO [22] plays the role of MEC Platform Manager, which has a global view of the ME Host, the available resources and the topology. This entity is responsible for deploying the network services and for providing orchestration functionalities. It handles requests from Operational Support Systems (OSS) for instantiating applications and, upon availability, it asks the VIM to allocate the virtual infrastructure and deploy the services. Within the VIM, Kubernetes is the container-based infrastructure manager, while the SD-RAN 5G-EmPOWER controller focuses on the RAN aspects [23].

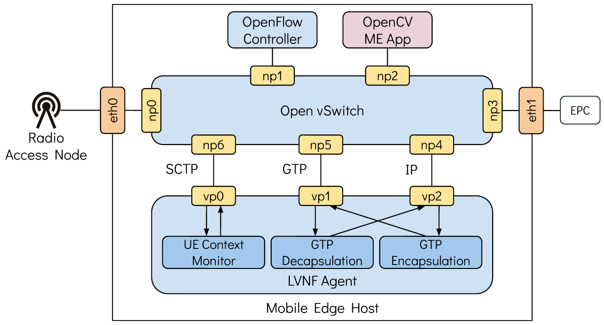


Fig. 2. ME Host architecture.

B. ME Host Deployment

The ME Host is based on a lightweight design leveraging virtualization technologies such as Docker containers and Click processes [24]. Figure 2 depicts the internal configuration of this entity. As can be observed, the traffic routing capabilities are provided by Open vSwitch, which is in turn operated by an OpenFlow controller managed by the SD-RAN controller through an intent-based interface. Conversely, a Click-based process, named Light VNF (LVNF) agent, analyzes the traffic between the radio access nodes and the EPC. The OpenFlow controller and the LVNF agent are executed as Docker containers, while other applications can run in additional containers or in external machines connected physically to the ME Host, thus ensuring scalability.

The traffic between the radio nodes and the EPC is intercepted by Open vSwitch for further processing by the LVNF agent as follows: (i) control plane traffic, which is encapsulated using the SCTP protocol, is steered to port `vp0`, (ii) user plane traffic, which is encapsulated in GTP packets, is forwarded to port `vp1`, and (iii) IP traffic is sent through port `vp2`. The SCTP packets in port `vp0` are used to gather context information from the User Equipment (UEs) and the GTP-U tunnels created to exchange traffic between such UEs and the core network. Later, these packets follow their path in the network. Conversely, GTP packets incoming to port `vp1`, in turn, are decapsulated using the aforementioned context information, and the underlying IP packets are delivered from port `vp2`. If the packet destination is one of the Apps within the ME Host, the Open vSwitch steers the IP packet to the corresponding virtual or physical port of such an App. Otherwise, they are redirected again to port `vp2` of the LVNF agent along with any other traffic originated. This IP traffic is then (re-)encapsulated in GTP packets and sent from port `vp1`, from which they can reach the mobile network.

As can be seen, the stateful decapsulation and encapsulation of GTP packets enables seamless communication between UEs and any service, regardless of whether it is placed as an App in the ME Host or in a cloud data center. Additionally, the monitoring of the SCTP traffic allows performing efficient session and mobility management of UEs. Moreover, these operations can be performed with no modifications to the protocol stack, making this solution completely vendor-agnostic.

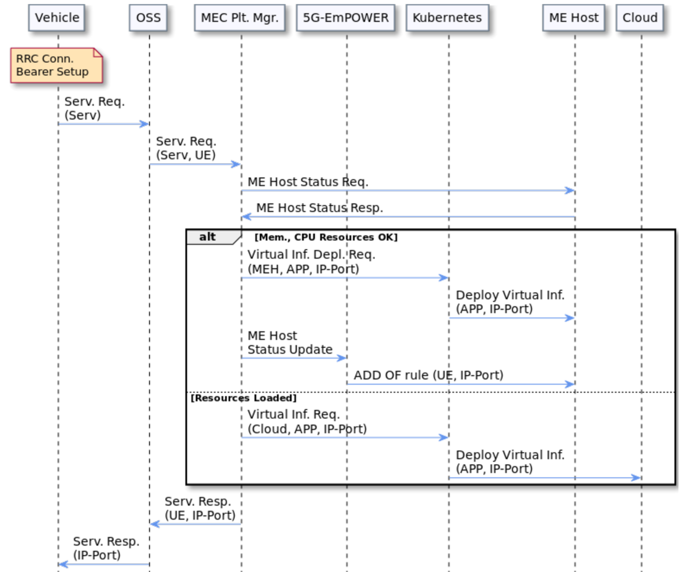


Fig. 3. Signaling for network service instantiation.

C. Network Service Instantiation

Figure 3 sketches the process to instantiate a network service. Note that for simplicity the signaling for bearers setup is omitted. In this scenario, the vehicles ask the OSS to deploy an assisted-driving service. This request is then forwarded to the MEC Platform Manager (`Serv. Req.` message).

Unless indicated otherwise, the Manager instantiates the service on the ME Host as long as there are enough resources (known from the `MEH Status` interchange). This decision is fed to Kubernetes, which prepares the virtual infrastructure to allocate the App on a given IP address-port. Then, the SD-RAN controller sets the OpenFlow rule needed to forward the traffic within the App (`ADD OF rule` message), and the MEC Manager provides the response to the OSS. By contrast, if the ME Host is saturated, an equivalent process is followed to deploy the service on the remote cloud.

D. Service Communication Model

Once the service is successfully deployed, traffic from vehicles containing road data is sent to the IP address-port provided, as shown in Fig. 4. Such traffic (`Proc. Req.` message) is intercepted by the ME Host, which, after inspection, checks the OpenFlow rules to determine whether the request must be handled on the ME Host or forwarded to the core network. In the former case, the traffic is processed by the corresponding App, and the obtained output is after encapsulated and sent back to the vehicle (`Processing Resp.` message). In the latter case, the traffic is reencapsulated and sent to the cloud for processing, which then returns the output to the radio node.

IV. ME App for Autonomous Driving

Among all CCAM applications, the lane line detection and the on-road object recognition algorithms represent the basis of

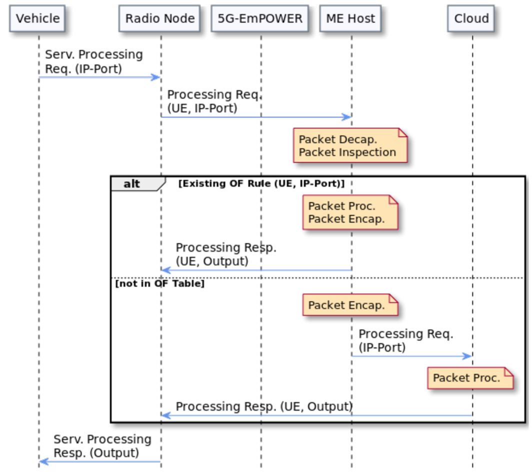
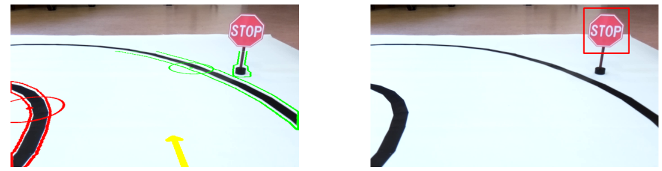


Fig. 4. Signaling for service communication and forwarding.



(a) Lane line detection. (b) On-road object detection.

Fig. 5. Deployed autonomous driving application.

Fig. 6. Location of the ME Host (green) and the remote clouds (red).

autonomous driving. With the aim of showing the capabilities of the presented platform in the V2X ecosystem, we have developed a simple application comprising the two aforementioned algorithms. This application processes the video stream fed by the vehicle using the OpenCV library, and returns the corresponding driving directions. The following subsections describe the details of these two algorithms.

A. Lane Line Detection

Lane line detection allows determining the path and the relative position of vehicles in the road. Therefore, the accuracy of this algorithm is key to ensure safe vehicle operation, and to coordinate the traffic in cooperative maneuvers. Nevertheless, there are many factors that may hinder this accuracy, such as adverse weather conditions, worn-out roads, or dirt. It is thus necessary to process the images to minimize errors.

After applying a Gaussian filter to reduce granularity, the images are transformed to the Hue, Saturation and Lightness (HSL) color space to extract the tones of the lane. The binarization of these pictures and the application of gradient filters, including Canny, results in the highlighting of the areas in the images that correspond to the lane lines. To determine the curvature of the lanes, the picture needs to be perspective-transformed on the basis of the parameters of the camera. Finally, the interpolation of the points conforming the lane lines are grouped and interpolated to estimate the curves. The result of this process can be seen in Fig. 5a.

Once the vehicle has been positioned in the lane, the problem of determining its path is simplified as the straight line with slope k equal to the derivative of the lane curves in the proximity of the vehicle. This straight line, l, is represented using its general equation:

$$l(x) = k \cdot x + b \tag{1}$$

Let (v_x, v_y) be the position of the vehicle, and (c_x, c_y) be the center of the circular trajectory of the vehicle that passes through point (v_x, v_y), it is satisfied that such a circumference is tangent to line l if the following condition is met:

$$\frac{(k \cdot c_x - c_y + b)^2}{k^2 + 1} = (c_x - v_x)^2 + (c_y - v_y)^2 \tag{2}$$

On the basis of this equation, it is possible to calculate point (c_x, c_y), and thus the radius of the trajectory.

B. On-Road Object Recognition

This algorithm involves the detection of entities within the view of the vehicle. As in the case of lane line detection, this algorithm involves some significant challenges, such as the understanding of text plates and the recognition of non-regulated signs. For the sake of simplicity, however, the implemented application only recognizes a limited set of traffic signs, resulting in the prediction shown in Fig. 5b.

Object classification is performed using the so-called Haar feature-based cascade classifiers, which have been widely used due to their ability to detect objects based on the features extracted from a set of positive and negative images [25]. Once the model is trained, it can be used off-line. Moreover, due to the layered modeling of features, the object detection process can be terminated prematurely, resulting in very low classification times. These properties make this classifier suitable for on-road object recognition.

V. Evaluation

A. Methodology

To evaluate the scenarios where computation needs to be offloaded while meeting certain latency requirements, such as the CCAM scenario, the above autonomous driving application has been tested under the architecture presented in Sec. III. In this regard, we measured different performance metrics, namely latency metrics and application metrics, in three locations: locally, in the edge, and in the cloud.

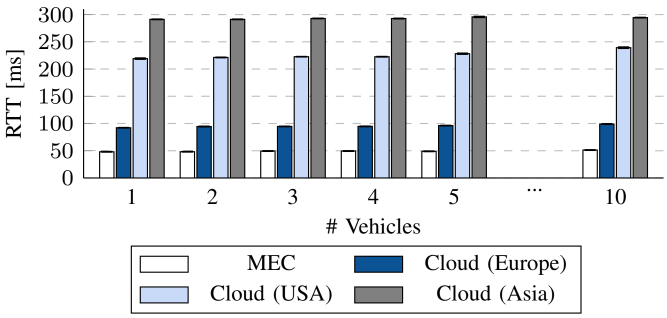


Fig. 7. RTT comparison between MEC and remote cloud in various locations.

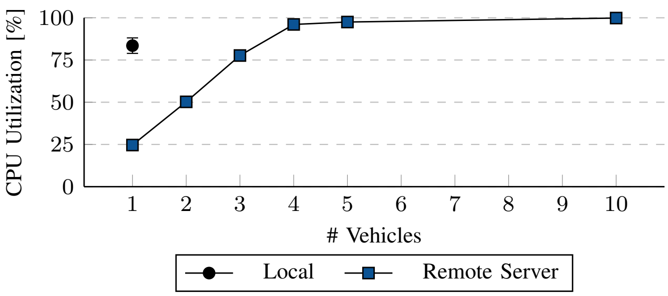


Fig. 8. CPU utilization comparison between local and remote processing for an increasing number of vehicle requests.

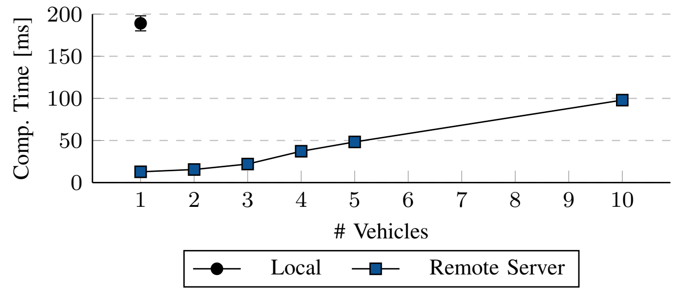


Fig. 9. Computation time per video frame and number of vehicle requests.

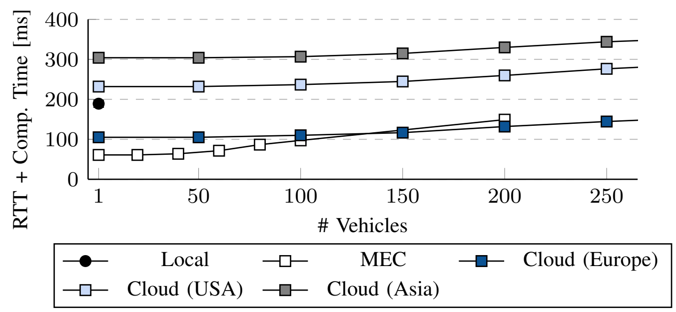


Fig. 10. Total average latency per video frame and number of vehicle requests in the proposed example scenario.

Regarding the latency assessment, we performed ping tests using ICMP messages to measure the Round-Trip Time (RTT), i.e., the time taken to receive a response to a vehicle request without considering any computation time. In this regard, 5 rounds of 1 minute long were performed for each test. In the case of the cloud, we selected three different locations, namely Europe, North America, and Asia, as shown in Fig. 6, while the vehicles and the ME Host were located in Italy.

For the evaluation of the application performance, we used instances of type *c4.4xlarge* available at the Amazon EC2 platform, which are composed of 16 virtual CPUs running on top of Intel Xeon E5-2666 v3 processors at 2.9 GHz, and 30 GB RAM memory. The platform selected as the embedded controlling system of the vehicle was composed of a quad-core 64-bit ARM Cortex A53 processor running at 1.2 GHz, 1 GB RAM memory, and LTE connectivity.

The analysis of the scalability of the proposed architecture in the edge and in the cloud was performed by measuring the influence of 1, 2, 3, 4, 5 and 10 simultaneous vehicles. Each vehicle transmitted live video encoded at 1,200 kbps and 30 frames per second, and received the output from the application each time a frame was processed.

B. Experimental Results

Figure 7 shows the average results obtained from the latency evaluation. As can be seen, there is a strong relationship between the RTT and the distance between the vehicle and the server. The latency involved by the ME Host is the lowest, 49.2 ms on average, given that it is located at the mobile edge, midway between the radio access node and the EPC. The latency of the cloud servers, however, is notably larger. In fact, the RTT is between 2 and 5 times higher. The number of concurrent vehicle queries also has a small effect on the latency in those cases in which the server saturates.

Regarding the application performance, Fig. 8 displays the CPU utilization of the application when executed locally, and when serviced remotely with respect to the number of vehicle queries. This utilization has the effect on the per-frame computation time shown in Fig. 9. As can be seen, the computation time increases to a larger extent when the CPU resources are exhausted, so the maximum number of concurrent queries that the server can attend depends on the requirements of the application. With respect to the memory usage, each instance uses approximately 162 KB RAM memory, which is negligible. The most significant conclusion drawn from these figures is, however, that the remote servers, i.e., the ME Host and the cloud, provide greater computational horsepower, and thus significantly lower computation times than the local processing approach. This highlights the important role of computation offloading in resource-constrained contexts.

Lastly, Fig. 10 plots the average time required to process each video frame sent to the application depending on the number of simultaneous vehicles connected, calculated as the sum of the RTT and the computation time, in an scenario where the ME Host and the cloud have different amount of resources. In particular, in this example we assumed that the ME Host can allocate 20 *c4.4xlarge* instances, whereas the cloud can hold 50 of these instances. With the aim of presenting a better view of the results, however, the maximum number of vehicles in the figure has been limited to 250.

As seen in the figure, cloud offloading can result in significantly larger delays compared with local processing in those cases in which the remote servers are located far away from the vehicles (i.e., USA and Asia). By contrast, cloud alternatives located closer to the vehicles provide acceptable latencies (i.e., Europe). It is MEC, however, the only approach that delivers suitable latencies for delay-sensitive applications, such as autonomous driving. Nevertheless, it should be noted that the average latency of the ME Host can be larger than that of the cloud depending on their respective loads at a given point in time. In this situation, the MEC Platform Manager can decide to allocate new requests into the cloud instead of the ME Host for latency minimization.

VI. Conclusions

This work aims at lowering the barrier for deploying autonomous and assisted driving applications and services in MEC environments. To this end, we leverage a lightweight MEC platform that converges SDN and NFV concepts into a single solution capable of supporting the tight latency and reliability requirements of this type of applications. Our solution builds upon lightweight computing and networking virtualization technologies such as Docker and Click. A proof-of-concept implementation has been introduced and validated in a practical use case, namely computer vision offloading. The results of the evaluation show that it is possible to use our platform to deploy computer vision applications. Moreover, this work allows to understand how to distribute the various elements of an autonomous and assisted driving application across vehicle, edge, and centralized clouds. As future work, we aim at extending this architecture by adding the ability to orchestrate and dynamically offload multiple service components characterized by different performance and latency requirements between MEC and cloud.

Acknowledgements

This work has been performed in the framework of the European Union's Horizon 2020 project 5G-CARMEN co-funded by the EU under grant agreement No 825012. The views expressed are those of the authors and do not necessarily represent the project. The Commission is not liable for any use that may be made of any of the information contained therein. In addition, this work has been supported by the Spanish Ministry of Science, Innovation and Universities (RTI2018-098156-B-C52, FEDER funds), and the Spanish Regional Government of Castilla-La Mancha (SBPLY/17/180501/000353).

References

[1] 5GAA, "An Assessment of LTE-V2X (PC5) and 802.11p Direct Communications Technologies for Improved Road Safety in the EU," Dec. 2017, White Paper.

[2] 3GPP, "TR 23.711. Enhancements of Dedicated Core Networks Selection Mechanism (Release 14)," Sep. 2016.

[3] A. Uçar, Y. Demir, and C. Güzeliş, "Object Recognition and Detection with Deep Learning for Autonomous Driving Applications," *Simul.*, vol. 93, no. 9, Sep. 2017.

[4] N. Gracias, R. Garcia, A. Shihavuddin, R. Campos, N. Hurtos, R. Prado, T. Nicosevici, A. Elibol, L. Neumann, and J. Escartin, *Computer Vision in Vehicle Technology: Land, Sea and Air*. Wiley, 2017.

[5] H. Wang, X. Li, H. Ji, and H. Zhang, "Federated Offloading Scheme to Minimize Latency in MEC-Enabled Vehicular Networks," in *2018 IEEE Global Communications Conference (GLOBECOM) Workshops*, Abu Dhabi, UAE, Dec. 2018.

[6] J. Ren, G. Yu, Y. Cai, Y. He, and F. Qu, "Partial Offloading for Latency Minimization in Mobile-Edge Computing," in *2017 IEEE Global Communications Conference (GLOBECOM)*, Singapore, Dec. 2017.

[7] R. Schmoll, S. Pandi, P. J. Braun, and F. H. P. Fitzek, "Demonstration of VR/AR offloading to Mobile Edge Cloud for Low Latency 5G Gaming Application," in *2018 IEEE Annual Consumer Communications & Networking Conference (CCNC)*, Las Vegas, NV, USA, Jan. 2018.

[8] S. K. Datta, R. P. F. Da Costa, J. Härri, and C. Bonnet, "Integrating Connected Vehicles in Internet of Things Ecosystems: Challenges and Solutions," in *2016 IEEE International Symposium on A World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, Coimbra, Portugal, Jun. 2016.

[9] J. Pillmann, B. Sliwa, J. Schmutzler, C. Ide, and C. Wietfeld, "Car-to-Cloud Communication Traffic Analysis Based on the Common Vehicle Information Model," in *2017 IEEE Vehicular Technology Conference (VTC Spring)*, Sydney, NSW, Australia, Jun. 2017.

[10] C. Xu, Y. Wang, Z. Zhou, B. Gu, V. Frascolla, and S. Mumtaz, "A Low-Latency and Massive-Connectivity Vehicular Fog Computing Framework for 5G," in *2018 IEEE Global Communications Conference (GLOBECOM) Workshops*, Abu Dhabi, UAE, Dec. 2018.

[11] A. Aissioui, A. Ksentini, A. M. Gueroui, and T. Taleb, "On Enabling 5G Automotive Systems Using Follow Me Edge-Cloud Concept," *IEEE Trans. Veh. Technol.*, vol. 67, no. 6, pp. 5302–5316, Feb. 2018.

[12] K. Zhang, Y. Mao, S. Leng, Y. He, and Y. Zhang, "Mobile-Edge Computing for Vehicular Networks: A Promising Network Paradigm with Predictive Off-Loading," *IEEE Trans. Veh. Technol. Mag.*, vol. 12, no. 2, pp. 36–44, Apr. 2017.

[13] L. Gillam, K. Katsaros, M. Dianati, and A. Mouzakitis, "Exploring Edges for Connected and Autonomous Driving," in *2018 IEEE Conference on Computer Communications (INFOCOM) Workshops*, Honolulu, HI, USA, Apr. 2018.

[14] J. Liu, J. Wan, B. Zeng, Q. Wang, H. Song, and M. Qiu, "A Scalable and Quick-Response Software Defined Vehicular Network Assisted by Mobile Edge Computing," *IEEE Commun. Mag.*, vol. 55, no. 7, pp. 94–100, Jul. 2017.

[15] ETSI, "MEC in 5G Networks," Jun. 2018, White Paper.

[16] 5GAA, "Toward Fully Connected Vehicles: Edge Computing for Advanced Automotive Communications," Dec. 2018, White Paper.

[17] F. Giust, V. Sciancalepore, D. Sabella, M. C. Filippou, S. Mangiante, W. Featherstone, and D. Munaretto, "Multi-Access Edge Computing: The Driver Behind the Wheel of 5G-Connected Cars," *IEEE Commun. Stand. Mag.*, vol. 2, no. 3, pp. 66–73, Oct. 2018.

[18] H. Chae, C. Mook Kang, B. Kim, J. Kim, C. Choo Chung, and J. Choi, "Autonomous braking system via deep reinforcement learning," in *2017 IEEE International Conference on Intelligent Transportation Systems (ITSC)*, Yokohama, Japan, Mar. 2017.

[19] T. Okuyama, T. Gonsalves, and J. Upadhay, "Autonomous Driving System based on Deep Q Learning," in *2018 International Conference on Intelligent Autonomous Systems (ICoIAS)*, Singapore, Oct. 2018.

[20] A. Uçar, Y. Demir, and C. Güzeliź, "Object Recognition and Detection with Deep Learning for Autonomous Driving Applications," *Simul.*, vol. 93, no. 9, pp. 759–769, Jun. 2017.

[21] T. Subramanya, G. Baggio, and R. Riggio, "lightMEC: A Vendor-Agnostic Platform for Multi-access Edge Computing," in *2018 International Conference on Network and Service Management (CNSM)*, Rome, Italy, Nov. 2018, pp. 198–204.

[22] R. Riggio, S. N. Khan, T. Subramanya, I. Grida Ben Yahia, and D. Lopez, "LightMANO: Converging NFV and SDN at the Edges of the Network," in *2018 IEEE/IFIP Network Operations and Management Symposium (NOMS)*, Taipei, Taiwan, Jul. 2018.

[23] E. Coronado, S. N. Khan, and R. Riggio, "5G-EmPOWER: A Software-Defined Networking Platform for 5G Radio Access Networks," *IEEE Trans. Netw. Service Manag.*, vol. 16, no. 2, pp. 715–728, Apr. 2019.

[24] E. Kohler, R. Morris, B. Chen, J. Jannotti, and M. F. Kaashoek, "The Click Modular Router," *ACM Trans. Comput. Syst.*, vol. 18, no. 3, pp. 263–297, Aug. 2000.

[25] P. Viola and M. Jones, "Rapid Object Detection Using Boosted Cascade of Simple Features," in *2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR)*, Kauai, HI, USA, Apr. 2001.