COPADRIVe - A Realistic Simulation Framework for Cooperative Autonomous Driving Applications

Bruno Vieira § , Ricardo Severino § ,Enio Vasconcelos Filho § , Anis Koubaa § *, Eduardo Tovar§
§Cister Research Centre
ISEP, Polytechnic Institute of Porto
Porto, Portugal
* Prince Sultan University, Saudi Arabia.
§(bffbv, enpvf, rarss, emt)@isep.ipp.pt, *akoubaa@psu.edu.sa

***Abstract*—Safety-critical cooperative vehicle applications such as platooning, require extensive testing, however, the complexity and cost involved in this process, increasingly demands for realistic simulation tools to ease the validation of such technologies, helping to bridge the gap between development and real-word deployment. In this paper we propose a realistic co-simulation framework for cooperative vehicles, that integrates Gazebo, an advanced robotics simulator, with the OMNeT++ network simulator, over the Robot Operating System (ROS) framework, supporting the simulation of advanced cooperative applications such as platooning, in realistic scenarios.**

***Index Terms*—C-ITS; Vehicular Platooning; ROS; OMNeT++**

I. Introduction

Modern embedded systems, coupled with the advancements of digital communication technologies, have been enabling a new generation of systems, tightly interacting with the physical environment via sensing and actuating actions: Cyber Physical Systems (CPS) [1]. These systems, characterized by an unprecedented levels of ubiquity, have been increasingly relying upon wireless communication technologies to provide seamless services via flexible cooperation, enabling true Systems-of-Systems (SoS).

Cooperative Vehicular Platooning (CoVP) is one of these emerging applications among the new generation of safety-critical Cooperating CPS. CoVP can potentiate several benefits, such as increasing road capacity and fuel efficiency and even reducing accidents [2], by having vehicle groups traveling close together. However, CoVP presents several safety challenges, considering it heavily relies on wireless communications to exchange safety-critical information, and upon a set of sensors that can be affected by noise. For instance, quite often in CoVP, wireless exchanged messages contribute to maintain the inter-vehicle safety distance, or to relay safety alarms to the following vehicles. Message losses or delays may lead to serious crashes among the vehicles in the platoon with dramatic consequences to them and to the remaining road members [3]. It is thus a truism, that timeliness and reliability of the communications are critic aspects for ensuring the safe operation of the platoon.

The ETSI ITS-G5 [4] is increasingly considered the enabler, ready-to-go communications technology for such applications, and although there has been extensive analysis of its performance [5]–[7], the understanding of its impact upon the safety of these SoS is rather immature. Hence, extensive testing and validation must be carried out to understand the safety limits of such SoS by encompassing communications. However, the expensive equipment and safety risks involved in testing, demands for comprehensive simulation tools that can as accurate as possible mimic the real-life scenarios, from the autonomous driving or control perspective, as well as from the communications perspective. The Robotic Operating System (ROS) framework is already widely used to design robotics applications, and aims at easing the development process by providing multiple libraries, tools and algorithms, and a publish/subscribe transport mechanism. On top of it, several simulation tools are capable to simulate the physics and several of the sensor/actuator and control components of these vehicles. On the other hand, several network simulators are available and capable of carrying out network simulation of vehicular networks. Nonetheless, these tools remain mostly separated from the autonomous driving reality, offering none or very limited capabilities in terms of evaluating cooperative autonomous driving systems.

In this work, we carried out the integration of a well-known ROS-based robotics simulator (Gazebo) with a network simulator (OMNeT++), by extending Artery [8], enabling a powerful framework to test and validate cooperative autonomous driving applications. In the one hand, we leverage upon Gazebo's robotic simulation most prominent features, such as its support for multiple physics engines, and its rich library of components and vehicles in integration with ROS, which enables us to build realistic vehicle control scenarios. On the other hand, OMNet++ supports the underlying network simulation relying on an ITS-G5 communications stack which is, currently, the *de-facto* standard for C-ITS applications in Europe. This integration provides the support for an accurate

This work was partially supported by National Funds through FCT/MCTES (Portuguese Foundation for Science and Technology), within the CISTER Research Unit (UID/CEC/04234); also by the Operational Competitiveness Programme and Internationalization (COMPETE 2020) under the PT2020 Partnership Agreement, through the European Regional Development Fund (ERDF), and by national funds through the FCT, within project POCI-01-0145-FEDER-032218 (5GSDN).

978-1-7281-0142-2/19/$31.00 ©2019 IEEE

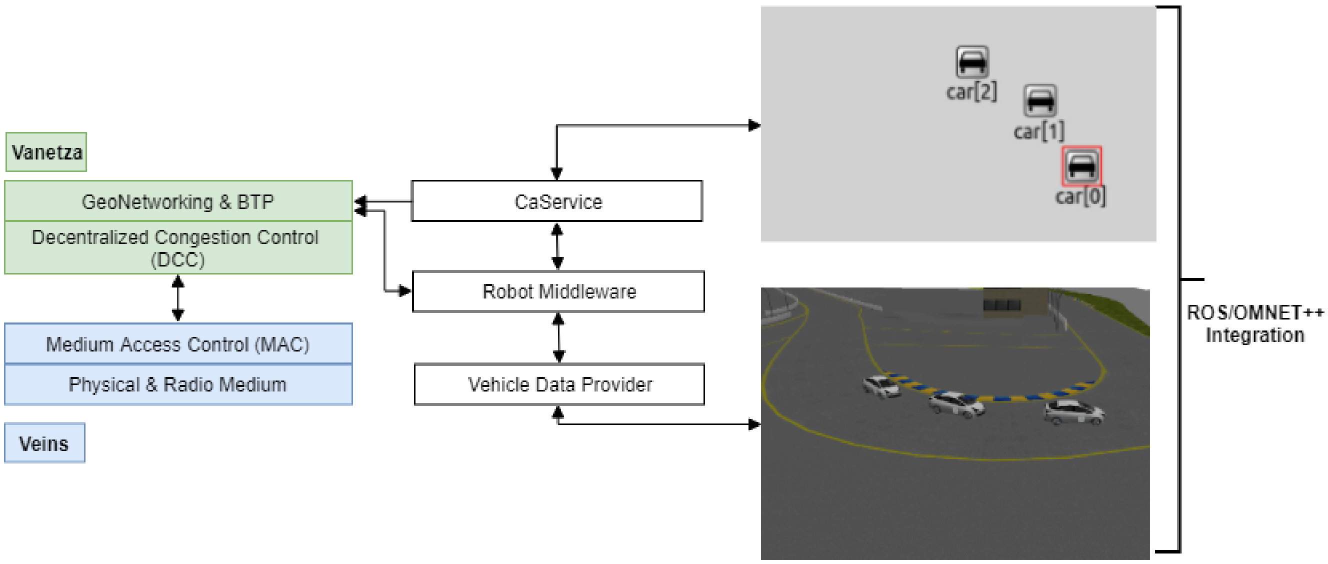

Fig. 1: Framework Architecture

analysis of the communications impact upon the cooperative application, and on the other hand, the tools to carry out a thorough evaluation of the network performance using the OMNet++/INET framework.

We present several contributions in this paper: (1) we implement a co-simulation framework for cooperative autonomous driving applications, relying on ITS-G5 communications and integrating other open-source tools with ROS support; (2) as a prove-of-concept, we implement a platooning control model, solely dependant on V2V communications, to assess the simulation framework, and (3) we analyse the impact of typical CAM settings, provided by the Basic System Profile (BSP) as standardized in ITS-G5 [4], upon the platooning behaviour, and evaluate its adequacy for a fully CoVP model.

In the remaining of this paper, in Section II we overview the related work, in particular the current state-of-the-art regarding connected vehicles simulation. In Section III, we describe the framework architecture, focusing on several implementation details. Finally, in Section IV we present different test scenarios and the experimental results, and conclude the paper in Section V.

II. Related Work

Currently, most relevant simulation frameworks focus on enabling an integration between traffic and network simulators to support the evaluation of Intelligent Traffic Systems (ITS). Some examples include iTETRIS [9] which integrates SUMO and ns-3, but it is pretty much stagnant, and unresponsive; VSimRTI [10] using an ambassador concept to support integration of virtually any simulator. Different traffic simulators and communication simulators have already been integrated, such as, the traffic simulators SUMO and PHABMACS and the network simulators ns-3 and OMNeT++; Artery [8], provides an integration of Veins (integrating SUMO traffic simulator and OMNet++) and the Vanetza ITS-G5 implementation. We identified Artery as the most mature project, which supported the best features to serve as guideline for our integration. In general, although these simulators may suffice to analyze macroscopic/mesoscopic, or even microscopic vehicular models, they are inadequate to support an evaluation of sub-microscopic models, which focus on the physics and particular characteristics of each vehicle. An exception is Plexe [11], an extension of Veins, which aims at enabling platooning simulation by integrating OMNeT++ and SUMO [12] together with a few control and engine models. However, it only enables the test of longitudinal platooning i.e., no lateral control and lacks support of a ITS-G5 communication stack. It is also limited in its capabilities to simulate a rich autonomous driving environment, when compared to robotic simulators, which primary objective is to mimic a realistic deployment environment to validate autonomous control models, encompassing sensors and actuators, rich simulation environments, as well as realistic physics models. There are other robotic simulators available that can support this kind of simulation. Perhaps the most prominent one, due its support of V2V communication via ns-3 is Webots [13]. However, the tool only very recently became open-source, and it does not support ROS out-of-the-box. Also, the ns-3 plugin does not support simulation of a ITS-G5 communications stack. Our proposed framework provides a clear advantage regarding the analysis of cooperative autonomous driving applications, in particular CoVP. It relies on Gazebo to enable a realistic simulation environment for autonomous systems via accurate modelling of sensors, actuators and vehicles, while harnessing the power of the ROS development environment, for developing new and complex algorithms from scratch using its ROS C++/Python framework.

III. Framework Architecture

A. OMNeT++'s Modules Overview

Our simulation framework was built over the Veins simulator and the Vanetza communications' stack implementation,

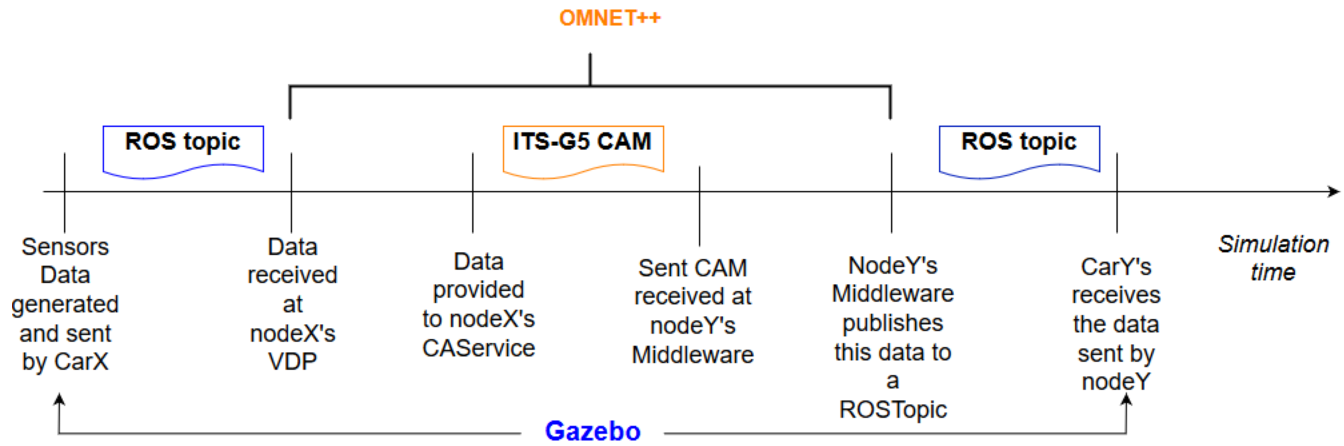

Fig. 2: Data workflow

borrowing and extending much of the middleware components from the Artery framework. It relies on ROS publish/subscribe mechanisms to integrate OMNeT++ with Gazebo, represented at Fig. 1. Each OMNeT++ node represents a car's network interface and contains a Vehicle Data Provider (VDP) and a Robot Middleware (RM). VDP is the bridge that supplies RM data from the Gazebo simulator. RM uses this data to fill ITS-G5 Cooperative Awareness Messages (CAM's) data fields (i.e Heading, Speed values) through the Cooperative Awareness Service (CaService) that proceeds to encode this data fields in order to comply with ITS-G5 ASN-1 definitions. RM also provides GPS coordinates to define the position of the nodes in the INET mobility module.

B. Synchronization Approach

OMNeT++ is an event-driven simulator and Gazebo a time-driven simulator, therefore synchronizing both simulators represented a key challenge. In order to accomplish this, a synchronization module was implemented in OMNeT++, to carry out this task, relying upon ROS "/Clock" topic as clock reference. The OMNeT++ synchronization module subscribes to ROS' "/Clock" topic, published at every Gazebo simulation step (i.e. every 1ms) and proceeds to schedule a custom made OMNeT++ message for this purpose ("syncMsg") to an exact ROS time, which allows the OMNeT++ simulator engine to generate an event upon reaching that timestamp and so to be able to proceed with any other simulation process that should be running at the same time (e.g. CAM generation by CAService).

C. Data Workflow

Fig. 2 presents a quick overview on how data flows from carX's sensors into carY's control application, working its way through Gazebo into OMNeT++ and then into other Gazebo's car following a CAM transmission between different nodes in OMNeT++. To note that nodeX and nodeY represent the network interface of both carX and carY, respectively.

To evaluate the framework's stability and limits, we analysed its inherent latency and/or computing delays. Fig. 4 presents the delay between an OMNeT++ node reception of a CAM (from a network transmission) and its reception by the Gazebo's vehicle model, after receiving it via ROS Pub/Sub mechanisms. Following Fig. 2 timeline, the timestamps recorded were taken at CAM reception at the node's Middleware and upon Gazebo's car application callback on this referred ROS topic, at different CAM sending frequencies (10, 5, 3.3 and 2.5 Hz). From what we can extract from the recorded data, this delay mostly coming from the ROS underlying Pub/Sub mechanisms, doesn't seem to be severely affected by the CAM sending frequency. Despite the maximum obtained delay slightly increased with traffic, the observed latency close to 0.25 milliseconds, is not sufficient to impact or compromise the application under test.

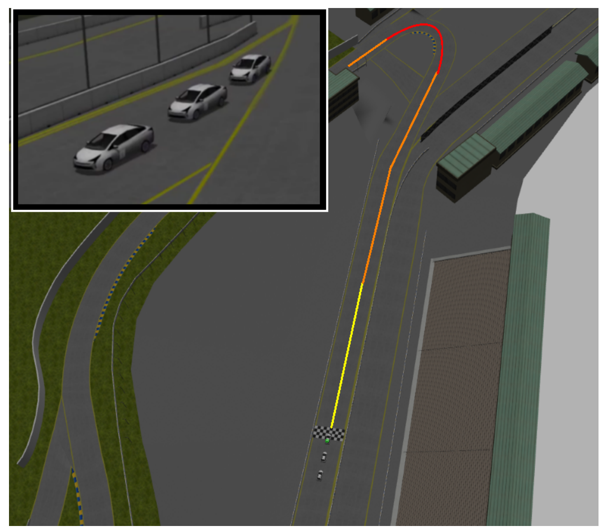

Fig. 3: Platooning Trajectory

IV. Experimental Results

The simulation is composed of three vehicles, modeled from a Toyota Prius, running a PID-based platooning control model [6] that solely relies on CAM messages to maintain the platooning service, with a safe distance set to 8 meters. The simulation results were extracted from 45-seconds long

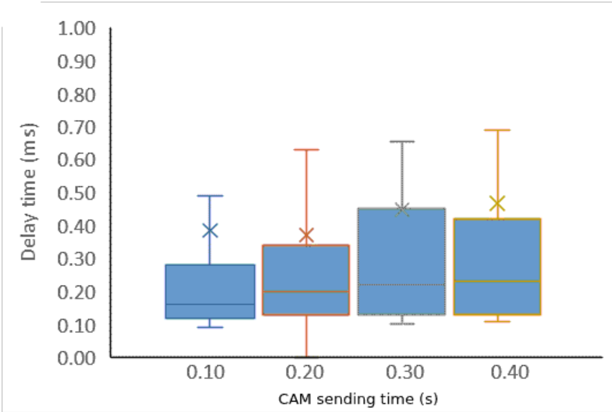

Fig. 4: CAM Exchanging Delay - Scenario A

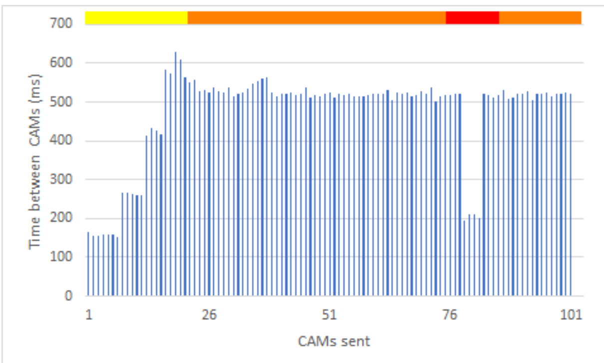

Fig. 5: Period CAM BSP

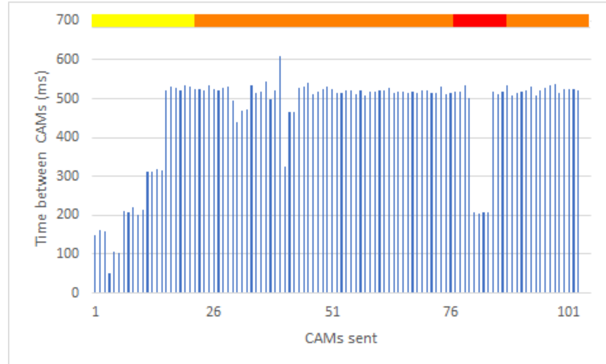

Fig. 6: Period CAM BSP for Platoon

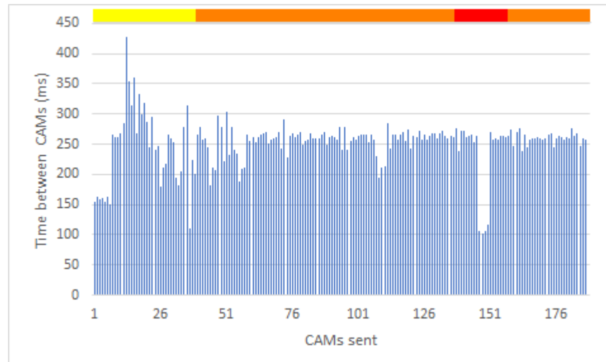

Fig. 7: Period CAM Custom SP

runs, in four different scenarios where the platoon safety was assessed: in scenario A, we setup fixed CAM frequencies, in scenario B the standard Basic System Profile (BSP) from ETSI [14] was used, in C we used the BSP with platooning-defined specifications [4] and in D we defined and evaluated customized settings for BSP. The simulation environment and simulated platooning trajectory in the 45-second run is represented in Fig. 3. The yellow line represents the initial acceleration path, where the follower car is still accelerating to reach the setpoint distance (8 meters) between itself and its leader. In orange, we represented the path in which the platooning is stable, and in red, a hard turn in which platooning behaviour is greatly dependant on the number of CAMs exchanged. The presented experimental results also contain this color reference to help relating them with the relevant portion on the track.

A. Scenario A: Fixed CAM frequencies

Four CAM sending frequencies were evaluated (i.e. 10, 5, 3.3 and 2.5 Hz), guaranteeing that at the highest CAM frequency, CAM messages will always be provided with fresh information. Fig. 8 shows the vehicle inter distance in each test and Fig. 10 presents the steering angles. We analyzed the impact of different CAM exchanging frequencies on the behavior of the second car, regarding the forward distance and steering angles to analyze how different CAM exchanging frequencies affected the CoVP control. The CoVP starts from parked position, and the follower only engages platooning after the leader starts moving forward, thus the follower needs to accelerate to catch up to its leader. It is also clearly noticeable that, for a CAM inter arrival time of 0.4 s, while approaching the left hard turn (in red), the follower lost track of the leader vehicle, making a full-stop. At higher CAM sending frequencies, we can observe that the CoVP PID controller shows better stability, and the inter-distance stability improves. These issues are also particularly visible regarding the steering behavior. For the first three CAM inter arrival times, the steering angles follow the leader's with a slight delay, which increases with frequency. For an inter arrival time of 0.4 s, the steering angles of the follower are no longer inline with the leader's (Fig. 10). CAM sending frequency is too low to keep the follower updated with leader's steering corrections, resulting in minimal or nearly non-existent steering inputs. Upon entering the left U turn, the follower's controller struggles to keep up with the steering of the leader, while it completely fails to do so for inter arrival times of 0.4 s. Among the several runs, at different frequencies, we notice a consistent behaviour, in such way that the higher the CAM sending frequency, the stable the PID steering control. However, fixing a CAM frequency represents a sub-optimal approach for CoVP, considering that excessive CAM traffic will often be generated, which can negatively impact the throughput of the network. With this in mind, ITS-G5 proposed BSP to dynamically trigger CAMs. In the following scenarios we evaluate its performance in the same context.

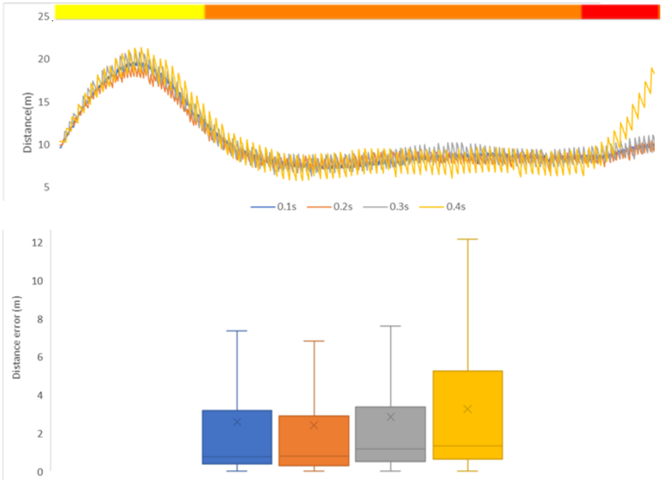

Fig. 8: Vehicle inter-distances - Scenario A

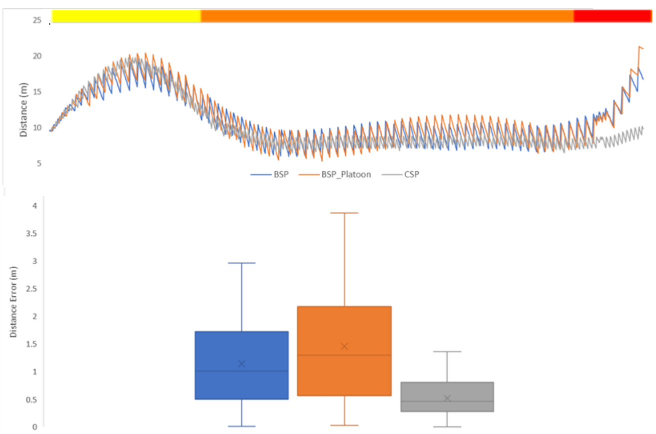

Fig. 9: Longitudinal distances analysis in different scenarios

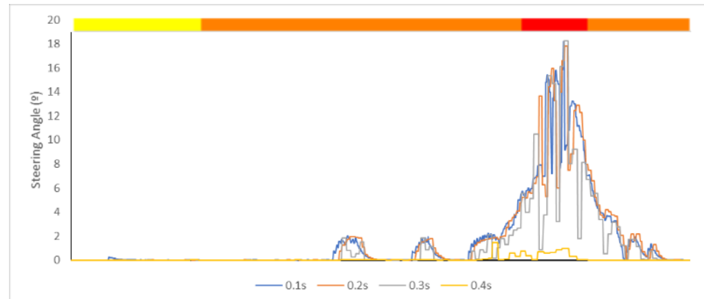

Fig. 10: Steering Angles - Scenario A

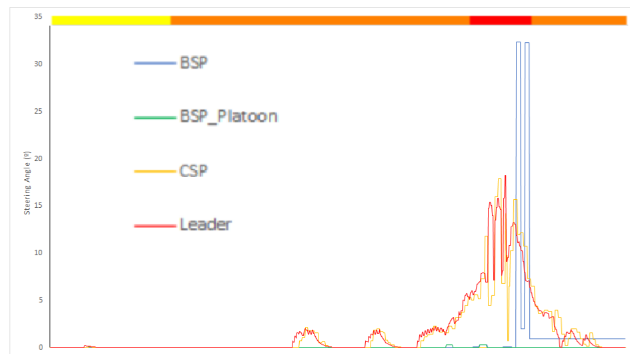

Fig. 11: Steering Analysis for different scenarios

B. Scenario B: Basic System Profile

For this Scenario we analyzed the BSP as standardized in ITS-G5 [14]. This profile defines the frequency at which CAMs should be triggered, considering vehicles' dynamics. BSP defines an interval of 0.1 seconds to 1 second between CAMs, except upon one of the following conditions, at which a CAM message must be immediately triggered [14]:

- the absolute difference between the current heading of the originating vehicle and the heading included in the CAM previously transmitted by the originating vehicle exceeds 4 degrees;
- the distance between the current position of the originating vehicle and the position included in the CAM previously transmitted by the originating vehicle exceeds 4 m;
- the absolute difference between the current speed of the originating vehicle and the speed included in the CAM previously transmitted by the originating vehicle exceeds 0,5 m/s.

CAM reception intervals are presented in Fig. 5. CAM sending frequencies approach 2.0 Hz, mostly due to the second triggering condition, since the CoVP speed during the orange straight part of the track is constant around 8 m/s. However, there are some high frequency triggers in the early iterations, resulting from the quick acceleration at the initial portion of the track, while trying to close the distance gap to the leader. It is also possible to observe that BSP triggers higher frequencies in response to the hard left turn (red portion of the track), that quickly shifts the leader's heading. Still, as observed in Figures 11 and 9 this increase in frequency was insufficient to maintain a stable CoVP control using this control model, and fails to follow leader's steering control. Therefore, we conclude that BSP is not well-tuned for more demanding CoVP scenarios, in which the control models exclusively rely upon cooperative support. While still trying to minimize network usage, and maintaining stable platooning control, we analyze and improve on the BSP settings in the following scenarios.

C. Scenario C: Basic System Profile for platooning

In this scenario, we analyze an extension to the ITS-G5's BSP specified in [4], which recommends improved BSP settings for platooning scenarios. One of its most significant changes, was to limit the minimum frequency between CAM transmission to 2 Hz, double of the one defined for the original BSP. Test results are quite similar to the usage of the original BSP settings, as triggering conditions remain the same. As depicted in Fig. 6 and similarly to scenario B, CAM inter arrival times remain around 2Hz, in this case as a result of the minimum frequency limit set. Concerning CoVP behaviour. Figures 11 and 9 depict a similar behaviour to scenario B, which results in a failure to execute the U turn. This is a consequence of platoon instability. Concerning distance error in regards to the set point, for instance, both scenarios B and

TABLE I: Comparison between Scenarios - Number of messages and Safety Guarantee

Scenario	Fixed Frequencies				BSP	BSP Plat.	CSP
	10	5	3.3	2.5			
Message	441	227	151	113	101	101	181
Safety	OK	OK	OK	NOK	NOK	NOK	OK

C, present similar and significant errors, resulting from low CAM update frequency.

D. Scenario D: Custom System Profile for Platooning

For this scenario we setup a Custom System Profile aiming at balancing the network load originated by CAM exchanging, while guaranteeing stability. With this in mind, our approach was to adapt the second CAM triggering condition mentioned at scenario B, by changing it to 2 meters instead of 4 meters. This change impacted the CoVP behaviour considerably, both in the number of CAMs sent and its frequency, as it's possible to check at Fig. 7. As shown in Figures 5 and 6, the CSP conditions caused CAM triggering to happen much more frequently than in previous cases, resulting on a more stable CoVP control when compared to scenarios B and C (Figures 11 and 9). This also translates into a significant decrease of distance errors to the leader, leading to a smoother control. In facto, only this changed enabled the CoVP to successfully complete the hard left turn (Fig. 3).

Table I presents the number of CAM messages sent during simulation for each scenario. As shown, a fixed frequency between 3.3 Hz and 2.5 Hz should be at the threshold borderline balance to maintain CoVP control. However, fixing this frequency is not the most reasonable approach since it can cause unnecessary CAM message transmissions. With this in mind, a System profile approach as defined in ITS-G5 should be the optimal way to handle this, however, as we were able to confirm with scenarios B,C and D this kind of profiling should be adapted to the use-case and particularly to the control model. For this particular control model under test, CAM information availability is crucial to maintain a stable behaviour. This kind of profiling can be easily carried out using our framework, by fully-specifying the simulation environment and CoVP control model over ROS/Gazebo, while using OMNeT++'s capabilities to analyze the network performance and to provide new extensions to the ITS-G5 communications stack, carrying out an integrated in depth analysis of CoVP behaviours.

V. Conclusions and future work

This paper proposes a sub-microscopic framework for cooperative driving simulation, integrating the Gazebo simulator and ROS robotics framework, with the OMNeT++ network simulator. Using this framework, as a preliminary proof-of-concept, we implemented and validated different scenarios to evaluate the behaviour of a CoVP control model, exclusively dependant on CAM exchanging. We analyzed the impact of different CAM exchanging frequencies and ITS-G5 BSP recommendations to validate the correctness of the simulation framework.

COPADRIVe successfully enabled this analysis, within a rich and realistic simulation environment, both from the control and communications perspective. We plan to complement these analysis with relevant network performance results and to increase the complexity of the scenario and CoVP control model. At the communications level we will be including external traffic sources to evaluate this and other CoVP models in congestion scenarios.

We firmly believe this tools has the potential to support advanced realistic ITS cooperative autonomous driving scenarios, and to help reducing technology validation effort and cost.

References

[1] D. Jia, K. Lu, J. Wang, X. Zhang, and X. Shen, "A Survey on Platoon-Based Vehicular Cyber-Physical Systems," *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 263–284, 2016.

[2] A. Talebpour and H. S. Mahmassani, "Influence of connected and autonomous vehicles on traffic flow stability and throughput," *Transportation Research Part C: Emerging Technologies*, vol. 71, pp. 143–163, Oct. 2016.

[3] P. Fernandes and U. Nunes, "Platooning With IVC-Enabled Autonomous Vehicles: Strategies to Mitigate Communication Delays, Improve Safety and Traffic Flow," *IEEE Transactions on Intelligent Transportation Systems*, vol. 13, no. 1, pp. 91–106, Mar. 2012.

[4] ETSI, "ETSI TR 102 638," ETSI, Tech. Rep. V1.1.1, 2009.

[5] O. Karoui, M. Khalgui, A. Kouba, E. Guerfala, Z. Li, and E. Tovar, "Dual mode for vehicular platoon safety: Simulation and formal verification," *Information Sciences*, vol. 402, pp. 216–232, Sep. 2017.

[6] O. Karoui, E. Guerfala, A. Koubaa, M. Khalgui, E. Tovard, N. Wu, A. Al-Ahmari, and Z. Li, "Performance evaluation of vehicular platoons using Webots," *IET Intelligent Transport Systems*, vol. 11, no. 8, pp. 441–449, Oct. 2017.

[7] Zhiwu Li, Oussama Karoui, Anis Koubaa, Mohamed Khalgui, Emna Guerfala, and Eduardo Tovar, "System and method for operating a follower vehicle in a vehicle platoon," Polytechnic Institute of Porto (ISEP-IPP), Portugal, Tech. Rep. CISTER-TR-181203, 2018.

[8] R. Riebl, H. Gnther, C. Facchi, and L. Wolf, "Artery: Extending Veins for VANET applications," in *2015 International Conference on Models and Technologies for Intelligent Transportation Systems (MT-ITS)*, Jun. 2015, pp. 450–456.

[9] M. Rondinone, J. Maneros, D. Krajzewicz, R. Bauza, P. Cataldi, F. Hrizi, J. Gozalvez, V. Kumar, M. Rckl, L. Lin, O. Lazaro, J. Leguay, J. Hrri, S. Vaz, Y. Lopez, M. Sepulcre, M. Wetterwald, R. Blokpoel, and F. Cartolano, "iTETRIS: A modular simulation platform for the large scale evaluation of cooperative ITS applications," *Simulation Modelling Practice and Theory*, vol. 34, pp. 99–125, May 2013.

[10] B. Schnemann, "V2x simulation runtime infrastructure VSimRTI: An assessment tool to design smart traffic management systems," *Computer Networks*, vol. 55, no. 14, pp. 3189–3198, Oct. 2011.

[11] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. L. Cigno, "Plexe: A platooning extension for Veins," in *2014 IEEE Vehicular Networking Conference (VNC)*, Dec. 2014, pp. 53–60.

[12] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y. Fltterd, R. Hilbrich, L. Lcken, J. Rummel, P. Wagner, and E. WieBner, "Microscopic Traffic Simulation using SUMO," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, Nov. 2018, pp. 2575–2582.

[13] I. Llatser, G. Jornod, A. Festag, D. Mansolino, I. Navarro, and A. Martinoli, "Simulation of cooperative automated driving by bidirectional coupling of vehicle and network simulators," in *Intelligent Vehicles Symposium (IV), 2017 IEEE.* IEEE, 2017, pp. 1881–1886.

[14] ETSI 2018, "ETSI EN 302 637-2," ETSI, Tech. Rep. V1.4.0, Aug. 2018.