IEEE Access
Multidisciplinary : Rapid Review : Open Access Journal
IEEE VEHICULAR TECHNOLOGY SOCIETY SECTION

Received 6 November 2024, accepted 25 November 2024, date of publication 27 November 2024, date of current version 9 December 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3508473

SURVEY

A Survey of Simulation Tools for Cooperative Positioning in Autonomous Vehicles

IVO SILVA[1,2], HÉLDER SILVA[1], FABRICIO BOTELHO[3], AND CRISTIANO PENDÃO[1,4]
[1]ALGORITMI Research Center, University of Minho, 4800-058 Guimarães, Portugal
[2]Center for Microelectromechanical Systems (CMEMS), University of Minho, 4800-058 Guimarães, Portugal
[3]Bosch Car Multimedia Portugal, S.A., 4701-970 Braga, Portugal
[4]Department of Engineering, University of Trás-os-Montes and Alto Douro, 5000-801 Vila Real, Portugal

Corresponding authors: Ivo Silva (ivo@dsi.uminho.pt) and Cristiano Pendão (cristianop@utad.pt)

This work is supported by national funds, through the Operational Competitiveness and Internationalization Programme (COMPETE 2020) [Project n° 179491; Funding Reference: SIFN-01-9999-FN-179491].

ABSTRACT Advanced Driver Assistance Systems (ADAS) and autonomous driving require high positioning performance (high accuracy, reliability, and availability). These requirements are not always possible due to disruptions and limitations on Global Navigation Satellite System (GNSS) signals. Cooperative positioning approaches aim to mitigate the drawbacks of GNSS by exploring the collaboration of road participants enhancing positioning performance. The development, testing and evaluation of cooperative positioning approaches is a complex process that is very difficult to conduct in real world since it depends on several vehicles equipped with sensors for perception and localization, as well as referencing systems which are expensive. To overcome this, researchers rely on simulation tools to experiment with their systems. This paper presents a study on simulation tools suitable for testing cooperative positioning scenarios using GNSS as one of the main sensors. We focus on tools capable of simulating realistic environments for vehicles with multiple sensors (autonomous driving), traffic and mobility models, V2X communications, and generating raw GNSS signals. We compare applications, features, and interfaces of these tools and analyze their integration for a cooperative positioning simulation pipeline.

INDEX TERMS Advanced driver assistance systems, autonomous driving, autonomous navigation, autonomous vehicle, cooperative positioning, collaborative positioning, GNSS, V2X communications, traffic and mobility models, simulation, simulator, synthetic data, survey.

I. INTRODUCTION

Global Navigation Satellite System (GNSS) based positioning provides accurate measurements in ideal conditions, i.e., when the signal is received with a direct Line-Of-Sight (LOS) to the satellite. Accuracy is typically within 1-3 m of error [1], [2] for standard positioning and can achieve decimeter-level accuracy using correction services like Precise Point Positioning (PPP) and Real-Time Kinematic (RTK) [3], [4]). Most receivers support multiple satellite constellations to increase the availability and reliability of the system. Despite that, there are still situations that cause disruptions in GNSS signals, namely, urban canyons where multiple buildings obstruct signals, limiting their reception due to propagation effects such as multipath, scattering, absorption, and also tunnels where GNSS signals are not available.

In vehicular scenarios, cooperative approaches involve road participants working together to determine their relative positions, thereby enhancing their absolute positions. Cooperative approaches [5], [6], [7], [8] appear as a solution to help in reducing errors from non-cooperative positioning methods such as GNSS [9]. Nowadays vehicles are equipped with several sensors such as, Radio Detection and Ranging (radar), Light Detection and Ranging (LiDAR), and cameras, as well as a means of communication enabled by Vehicle-to-Everything (V2X) communication interfaces, which enable vehicles to operate in a cooperative way, sharing sensor data to improve the overall positioning performance.

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

Cooperative positioning approaches are quite diverse, with researchers exploring various methods. For example, in [5], vehicles exchange raw GNSS measurements to estimate relative positioning in a cooperative way. Another approach is demonstrated in [10], where the vehicle's relative positions are determined using Ultra-Wideband (UWB) ranges from two radios on two vehicles following each other. In dynamic urban scenarios with urban canyons, vision-based overlapping area detection and inter-ranging measurements are used collaboratively to improve positioning, as described in [11].

One of the main challenges in researching and developing cooperative positioning solutions is obtaining data from multiple sensors and vehicles simultaneously, along with highly accurate ground truth. These types of experiments are highly complex to conduct in the real world, mostly due to economic resources in terms of required materials (vehicles, sensors, reference systems, etc.) and human effort. Besides, repeating these experiments in different scenarios to assess cooperative systems under different conditions would be even more complex and difficult to achieve. Simulation provides a solution to these challenges as it supports multiple vehicles, each equipped with sensors (e.g., LiDAR, radar, GNSS, etc.), collecting data that can be used to test and evaluate cooperative systems. Also, most simulation tools offer the ability to simulate different scenarios (cities, motorways, etc.) and conditions (diverse weather conditions, high/low traffic, with/without pedestrians, etc.).

With the recent technological advancements in computer graphics, both on hardware and software, simulation has gained interest from the industry and academia to enable the development of new solutions for the automotive field. In the automotive context, there are simulation tools with different purposes and applications. Driving simulators replicate driving experience where drivers interact in real-time with a simulated environment, through physical controls, such as a steering wheel, pedals, and gear shift. These simulators are mostly used for driver training and testing, development of human-machine interfaces, and driver behavioral analysis. Examples of driving simulators include VI-Grade [12], SimDriver [13], STISIM Drive [14]. Autonomous driving simulators are typically software-based tools that focus on the development, testing and validation of autonomous driving and ADAS, sensor development, and development of positioning algorithms. These simulators allow high-fidelity physics and realism for 3D environment reproduction, supporting several sensors (e.g., LiDAR, radar, cameras, GNSS, among others), traffic generation, and often provide a way to integrate with real-world autonomous driving hardware through Hardware-In-the-Loop (HIL). CARLA [15], AWSIM [16], IPG Automotive CarMaker [17], AVSimulation SCANeR [18] are some of the most well known autonomous driving simulators. Traffic and mobility simulators such as SUMO [19] or Vissim [20], model the traffic, temporal and spatial mobility of vehicles in different conditions by implementing realistic mobility models. These simulators are often combined with network simulators, that simulate vehicular communications, e.g., OMNeT++ [21] or Network Simulator (NS-3) [22], which implement V2X communication protocols to emulate real communication between vehicles.

Simulation tools are practical because they easily create scenarios with different conditions, allowing the testing of various aspects in a controlled environment. They also offer a significant advantage by reducing the costs and time associated with real-world testing. Furthermore, these tools provide an opportunity for early-stage testing of algorithms and prototypes, before moving to real-world implementation.

In this paper, we present an overview of several simulation tools for automotive applications with an emphasis on the tools that can be used for the development and evaluation of vehicular cooperative positioning applications. The contributions of this paper are summarized as follows:

- Overview of cooperative positioning methods and simulation requirements;
- Comprehensive study of the state-of-the-art about software simulation tools divided into four categories: vehicle multi-sensors in a 3D realistic environment; traffic and mobility; V2X communication; and, raw GNSS measurements;
- Detailed comparison between simulation tools including their main application and features.
- Analysis of the methods that integrate multiple simulation tools to simulate cooperative positioning scenarios.

The remainder of this paper is organized as follows. Section II presents the related work. An overview of the cooperative positioning methods and requirements is made in Section III. Section IV presents the study on the state-of-the-art simulators, separated into four categories, as mentioned above. Section V describes several examples about the usage of simulation tools for cooperative positioning. Section VI summarizes the simulation tools analyzed and provides a detailed discussion about the functionality and applications of different types of simulators. Finally, conclusions and future work are presented in Section VII.

II. RELATED WORK

Several works have explored the use of simulation tools for vehicles, with particular focuses on unmanned vehicles, autonomous driving, vehicular networks, and cooperative positioning systems.

Craighead et al. [23] completed a survey of computer-based simulators for unmanned vehicles, analyzing both commercially available and open-source simulators, including aircraft simulators, robot simulators, and game engines. The authors concluded that the already available simulators provide the functionality needed for researchers working in this area, so they can use them instead of devising new ones. Similarly, Cook et al. [24] conducted a survey focused on simulators for autonomous underwater vehicles and robots, concluding that existing software is suitable for underwater vehicle simulations as well. In comparison with [23] and [24], this work focuses on both commercially available and open-source simulators for automotive applications, instead of simulators for unmanned vehicles and underwater vehicles.

A study about simulators for autonomous vehicle research was introduced in [25]. First, it analyzes sensors involved in autonomous driving, namely the environment perception sensors (e.g., ultrasonic, radar, LiDAR, cameras, etc.) and sensors used for positioning (e.g., GNSS, RTK, Inertial Measurement Unit (IMU), etc.). Then, the paper describes simulators for model-based development, the main game engines that can be used for simulation, the simulators for the robotics fields and simulators especially devised for autonomous vehicles. Since this paper was published, several simulation tools were made available which can be used nowadays. A similar survey was proposed by Kaur et al. [26]. Initially, the authors identified the main requirements for autonomous driving. Then, they compared several simulators, namely, MATLAB/Simulink, CarSim, PreScan, Gazebo, CARLA and LGSVL, aiming to study their capabilities in features such as perception, localization, and vehicle control, among others. The work in [27] provides a comprehensive survey on simulators for autonomous driving, categorizing simulators according to their applicability: traffic flow simulator – simulate movement of vehicles and other dynamic traffic participants within a transportation system; sensory data simulators – generate highly realistic sensory data and ground truth for perception tasks; driving policy simulator – generates executable traffic scenarios for development and evaluation of driving policy; vehicle dynamics simulator – replicates the dynamic behavior of vehicles based on physics principles; comprehensive simulator – is the integration of the functionalities listed prior to this, hence, a comprehensive simulator can support multiple tasks across the perception, planning, and control parts of autonomous driving systems (e.g., CARLA [28] or AirSim [29]). Then the survey discusses simulator fidelity regarding sensory data and vehicle dynamics. It does a thorough analysis on open-source simulators and identifies critical issues and how they can be addressed. Unlike previous works [25], [26], [27], this study specifically addresses the challenge of simulating a cooperative positioning scenario. It emphasizes the need to integrate multiple types of simulators to accurately represent aspects such as realistic environment, vehicle motion, sensors, and vehicular communications.

Numerous studies have been also made about simulation tools for Vehicular Ad-Hoc Networks (VANETs). In [30] the analyzed tools were separated into three categories, namely, mobility generators, network generators and VANET simulators. Mobility simulators model the real behavior of vehicles in traffic, network simulators emulate the exchange of messages among connected nodes, and VANET simulators mimic the characteristics of the VANET including mobility features such as motion constraints, traffic dynamics, real traffic scenarios, and the properties of the wireless communication networks. An updated review of VANET simulators in Weber et al. [31] describes the main features, supported technologies, and the capabilities to model safety, security and privacy functionalities which have motivated extensive research in VANETs over the last years. The comprehensive review of VANET simulators in [32] describes several tools that integrate the network simulator and mobility simulator to simulate VANET characteristics, classifying them as cross-coupled or tightly coupled according to the type of communication integrated simulators use. More recently, a systematic review simulators for VANETs was made in [33], to identify the applicability and availability of existing VANET simulators, mobility generators and network simulators. This study included simulation tools published in papers after 2015 and compared them in terms of provided features, type of software license and programming language of each tool. The main conclusions from the authors are that NS-3 and SUMO are the best choices for real-time VANET modelling, with NS-3 providing network simulation and SUMO providing mobility simulation. While the surveys in [30], [31], [32], and [33], address several aspects of VANET, this paper provides an analysis specifically from the perspective of cooperative positioning simulation, in which VANET simulators need to be integrated with other types of simulators, to accurately represent cooperative scenarios.

A survey on cooperative positioning using GNSS measurements is proposed in [34]. It provides an overview of the state-of-the-art works on GNSS-based cooperative positioning. As GNSS faces challenges in urban scenarios, such as Non-Line-Of-Sight (NLOS), multipath, cycle slips, among others, the papers details the systems proposed by the research community exploring vehicular networks within urban environments, where vehicles exchange GNSS data for improved positioning. Since most cooperative positioning solutions rely on GNSS-based techniques, integrating GNSS simulators is essential for realistic cooperative positioning scenarios. Hence, this paper includes an analysis of GNSS simulators capable of generating raw measurements considering error sources like thermal noise, atmospheric effects, and multipath.

Unlike other studies that focus on unmanned vehicles, autonomous driving, VANETs, mobility, or GNSS-based cooperative positioning, this paper offers a comprehensive survey of simulation aspects related to cooperative positioning. We begin with an overview of cooperative positioning and its applications in the automotive sector. Then we detail simulation tools, namely the realistic 3D simulators for autonomous driving, which support multiple sensors. We also cover traffic and mobility simulators, vehicular communication simulators, and GNSS data simulators. Additionally, we review several studies that propose simulation tools for cooperative positioning and driving scenarios, including those that integrate multiple tools for a realistic simulation of all these aspects.

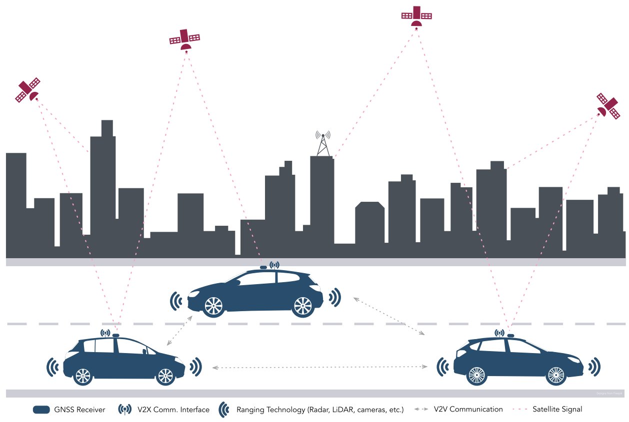

FIGURE 1. **Cooperative Positioning scenario: vehicles with GNSS share data between each other using V2X communication.**

III. COOPERATIVE POSITIONING OVERVIEW

Cooperative or collaborative approaches consist of multiple entities working towards a common goal, usually sharing sensor measurements or position estimates between actors to determine a cooperative position estimate. Fig. 1 illustrates an example of a cooperative positioning scenario where vehicles communicate with each other, sharing data to improve their overall position. In this scenario, GNSS provides the absolute positioning, which is prone to errors especially in urban canyon scenarios. Since the vehicles are equipped with several sensors, e.g., radar, LiDAR, and cameras, as well as a means of communication enabled by V2X communications interfaces, it enables them to operate cooperatively, sharing sensor data to enable relative positioning, and ultimately improving the overall positioning performance.

According to [6], cooperative approaches can be divided into four types: transponder-based ranging systems, in which vehicles estimate their relative position directly at the RF-signal level; GNSS reference-based localization techniques, where GNSS-based information is exchanged between vehicles using a dedicated communication technology; overlapping area detection-based, where vehicles share the perception of the environment using vision-based sensors to detect their relative positions; GNSS and cellular radio network-based, where vehicles share cellular network ranges for relative and absolute positioning which is combined with GNSS.

A. TRANSPONDER-BASED RANGING SYSTEMS

Different approaches and technologies may be used to determine range measurements between vehicles, which are then used to determine the relative positions of vehicles. For instance, in [35], Received Signal Strength Indicator (RSSI)-based radio ranging is used to obtain inter-vehicle distance estimates, which are used to localize a vehicle among its neighbors. UWB-based radio ranging is also effective for measuring short to medium distances (from 10 cm to 300 m), thus being also a suitable technology for cooperative positioning [10].

B. GNSS REFERENCE-BASED

Cooperative positioning based on GNSS presumes a communications link between vehicles, hence Vehicle-to-Vehicle (V2V) communication protocols are used to exchange information. There are different methods of localizing a vehicle using GNSS as a reference [34], using different architectures (centralized vs distributed), exchanging different types of data (GNSS position, pseudoranges, carrier-phase or Doppler), which can be combined with other measurements (e.g., IMU, odometer, vision, among others).

A distinction between absolute positioning and relative positioning must be made, since absolute positioning refers to the vehicle's position relative to Earth, whereas relative positioning has the objective of determining the relative position of a vehicle with respect to one or more other vehicles.

Cooperative approaches for determining absolute vehicle positioning-geographic location determined in reference to a global coordinate system—have been proposed in [9], [36], and [37]. These methods involve exploring GNSS measurements from cooperating vehicles for positioning algorithms and utilizing inter-vehicle radio ranging techniques. Through this cooperative process, pseudoranges are adjusted to estimate an absolute position.

GNSS-based relative positioning has been used to extend the limited perception of the vehicle's onboard sensors while providing better relative position and velocity estimates [38], [39]. These approaches are mostly based on sharing pseudoranges over V2V communication, but there are also other approaches where data from other sensors are also exchanged between vehicles.

C. OVERLAPPING AREA DETECTION BASED

This approach uses exteroceptive sensors to identify overlapping areas between vehicles and employs loop closure detection and factor graph optimization to estimate their relative pose (position and orientation relative to local frame of Visual/Inertial Navigation System (VINS)). In [11], this approach is explored with vision-based overlapping area detection and inter-ranging measurements for enhancing positioning in challenging outdoor urban scenarios. The approach sends feature descriptors and 3D landmark positions to a central station instead of raw image frames, which are used for overlapping area detection using loop closure detection in visual Simultaneous Localization and Mapping (SLAM). The relative pose of the vehicles is estimated using the Perspective-n-Point algorithm [40], which involves determining the position and orientation of a calibrated camera based on a set of n known 3D points in the world and their matching 2D projections in the image. However, this method requires a centralized station for receiving and processing visual information, which is a drawback.

D. GNSS SIGNALS AND CELLULAR RADIO NETWORK

Cellular radio network signals are commonly detected in urban environments, especially in high-density areas with many buildings. With the deployment of 5G, it is expected a high density of base stations to provide coverage of mm-Wave bands which require a line-of-sight. This signal abundance arises as another alternative for cooperative positioning. Cellular networks have also been explored for the integration of GNSS and 5G networks aiming to increase the availability and reliability of the vehicle's positioning system since 5G signals can be explored for positioning [41], [42].

E. COOPERATIVE POSITIONING SIMULATION REQUIREMENTS

Although cooperative positioning approaches vary considerably in their methods—whether transponder-based ranging systems, GNSS reference systems, overlapping area detection, or GNSS signals and cellular networks—the fundamental requirements for simulating any cooperative positioning scenario are as follows:

- **Realistic Environment** – Simulate a realistic environment with several vehicles and possibly other actors, such as pedestrians or cyclists. The 3D environment should be of high fidelity, incorporating detailed physics to accurately emulate the dynamics and interactions of vehicles, pedestrians, and other road users. This includes terrain, weather conditions, and various surface types to ensure comprehensive scenario testing.
- **Sensors Data** – Generate synthetic sensor data from several types of sensors, e.g., GNSS, LiDAR, radar, cameras, among others. The simulation should support multiple sensors for perception and positioning, enabling the development and testing of autonomous driving capabilities as well as cooperative positioning.
- **Traffic and Mobility** – Simulate mobility and traffic behavior of vehicles, which involves creating models that reflect real-world vehicle movements and traffic scenarios, such as urban traffic, highway driving, and congestion. The simulation should consider driver behaviors, traffic signals, and road regulations to replicate how vehicles navigate through different traffic conditions in a city.
- **Vehicular Communications** – Simulate a communication means to enable sharing of data between cooperative positioning actors. This includes simulating V2X communication technologies such as 5G and WAVE (Wireless Access in Vehicular Environments). The simulation should model the propagation channels, sources of errors, radio signals, and communication protocols involved, ensuring that vehicles can exchange information about their positions, movements, and sensor readings, in realistic conditions.
- **Ground Truth** – Provide ground truth information for simulated vehicles. Having accurate reference data is crucial for evaluating the performance of developed methods. This includes precise position, velocity, and orientation information, which can be used to assess the accuracy and reliability of the cooperative positioning systems under test.

Based on these requirements, the following section includes an analysis of the simulation tools necessary to simulate a cooperative scenario.

IV. SIMULATORS

Figure 2 summarizes the main components needed for simulating a realistic cooperative positioning scenario, according to the requirements specified in Section III-E. These components include a realistic 3D environment in which vehicles operate, featuring roads, buildings, and obstacles, and are equipped with various sensors for autonomous driving (e.g., LiDAR, radar, IMU, etc.) (Fig. 2 A). Another essential component is the traffic and vehicle mobility models, which replicate realistic vehicle behavior (Fig. 2 B). Besides, vehicular communications are considered, including the network protocols and the propagation channel that influences communication (Fig. 2 C). Accurate GNSS simulation is also included, providing raw GNSS pseudoranges and carrier phases while accounting for noise sources such as atmospheric effects and multipath (Fig. 2 D). Since no single tool can simulate all aspects of cooperative positioning, different simulators are designed to focus on specific elements depending on the use case.

In the following sections, we analyze several tools capable of simulating each of the components mentioned above. In Section IV-A, we review simulators that replicate realistic environments where vehicles, pedestrians, and other elements closely resemble the real world, to support the development, training, and validation of autonomous urban driving systems. These simulators are typically built on gaming engines such as Unity or Unreal Engine and include sensor models to emulate sensors such as cameras, LiDAR, radar, among others. In Section IV-B, we analyze intermodal traffic simulators that simulate the complex dynamics of urban mobility, integrating various transport modes like cars, bicycles, and pedestrians. These simulators are suitable for applying traffic models in cooperative scenarios. In Section IV-C, we examine communications simulators capable of modelling vehicular communications V2X, as in, Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I). These tools model the network protocols, propagation effects, and physical environments required for vehicular communication systems. The integration of communication capabilities is essential for testing cooperative driving scenarios, where vehicles share information such as speed, location, road conditions, and other sensor data to enhance safety and traffic flow. In Section IV-D, we analyze simulators that generate synthetic GNSS measurements. These tools recreate not only the satellite signals but also the effects of environmental conditions and receiver characteristics, resulting in realistic pseudorange measurements. This is critical for testing the

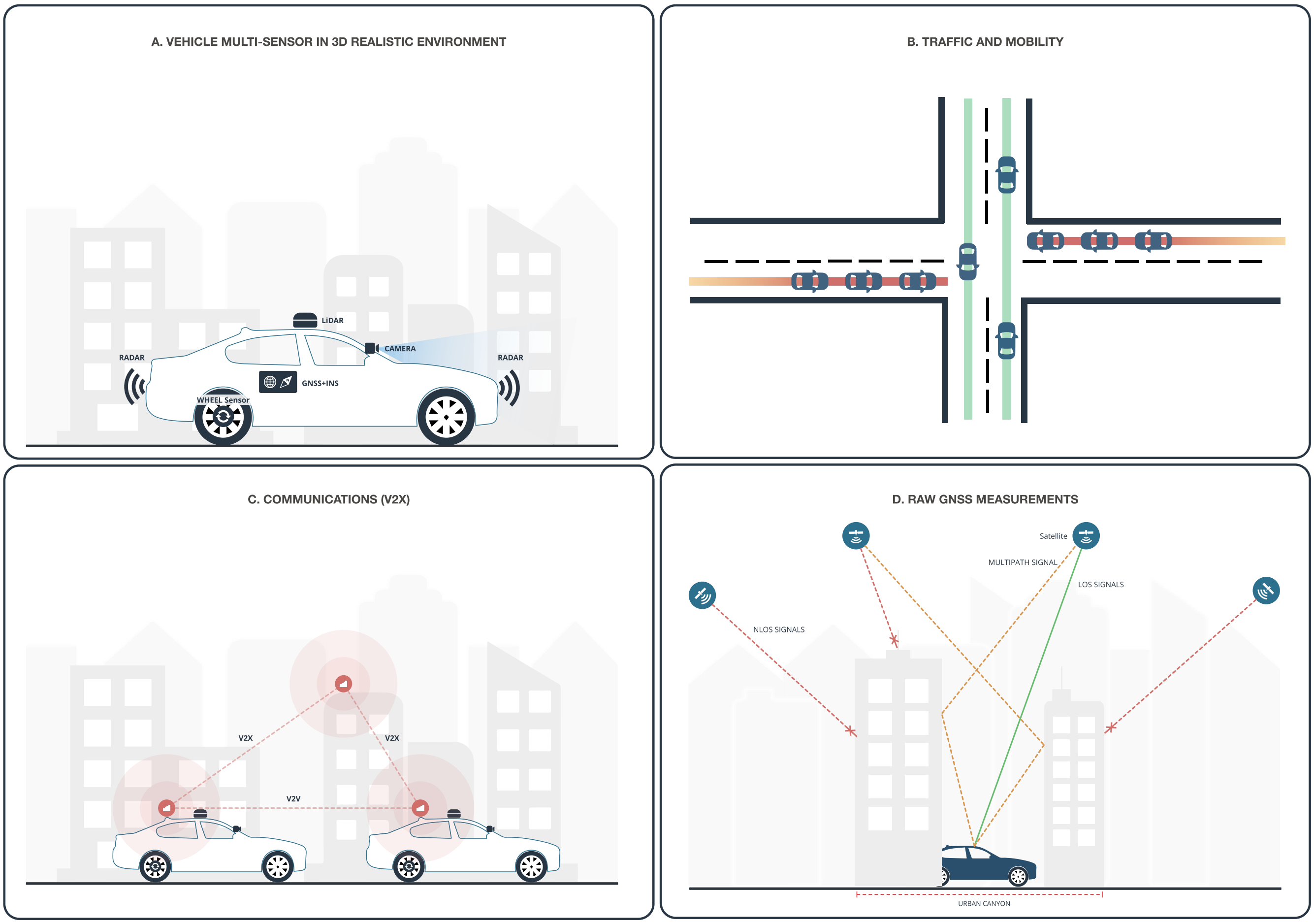

FIGURE 2. Cooperative positioning simulation components: A) realistic 3D environment with vehicles equipped with multiple sensors; B) vehicle's traffic and mobility models; C) vehicular communications (V2X) with realistic propagation channel and networking protocols; D) raw GNSS measurements considering satellite's orbits, atmosphere effects and propagation effects.

performance of cooperative positioning systems in scenarios where GNSS signals may be obstructed or degraded, e.g., in urban canyons or tunnels. Integrating these GNSS capabilities with other simulation types, such as traffic and communication systems, enhances the overall realism and accuracy of the simulation, allowing for comprehensive testing of vehicle performance in complex environments.

Despite the existence of many commercial solutions, we mostly focused on free and open-source tools, due to two reasons. First, they are free to use and do not require a paid license. Second, being open-source means that the code is open and available, allowing the community to extend and support new features.

A. VEHICLE MULTI-SENSORS IN 3D REALISTIC ENVIRONMENT

Multi-sensor simulators are mostly used to simulate autonomous vehicles scenarios, to test and develop new solutions for ADAS and autonomous navigation. These simulators generate synthetic data from vehicles equipped with various sensors, including radar, LiDAR, IMU, and GNSS. For autonomous vehicles, where decisions rely heavily on sensor data, it is essential that simulation platforms accurately replicate a 3D environment, with realistic elements like roads, trees, buildings, vehicles, etc. This high-fidelity environment is crucial, as it ensures that the data generated for exteroceptive sensors, such as cameras and LiDAR, accurately represents the complex, dynamic nature of real-world surroundings.

These simulators are based on game engines capable of emulating a 3D environment whose visuals and physics are similar to the real world. Some allow to set different weather conditions thus allowing to test challenging conditions for validation of sensors like cameras. Many of these simulators support both Software-In-the-Loop (SIL) and HIL. SIL involves integrating and testing the software components in a virtual environment to validate their behavior and performance before deployment on actual hardware. On the other hand, HIL, involves integrating actual hardware components into the simulation loop to test the hardware's interaction with the simulated environment. This is particularly useful for validating the performance and reliability of physical components in a controlled and repeatable manner. Co-simulation is another feature that multi-sensor simulators support, which allows two or more simulators to exchange data and interact in real-time. Vehicle multi-sensor simulators are often used in co-simulation with other simulators, for instance, generating traffic and mobility [43], or providing V2X capabilities [44], [45]. Additionally, Robot Operating System (ROS) integration through communication interfaces is often supported by these simulators, allowing to connect multi-sensor simulators with the ROS ecosystem which provides packages and tools for SLAM, navigation, and perception.

1) CARLA

CARLA [15], [28] was built from the ground up to support the development, training, validation, and testing of autonomous driving systems. It is an open-source simulator that provides open digital assets (urban layouts, buildings, and vehicles), that can be used in the simulated scenario. Some of the features this simulation platform supports include different types of sensors, environmental conditions, complete control of all static and dynamic actors, and maps generation, among others. The API provided in CARLA enables users to control many aspects related to the simulation, including the weather, traffic generation, types of vehicles, pedestrian behaviors, sensors, and other aspects. To enable dynamic movement of vehicles, CARLA has a traffic manager module that controls vehicles in autopilot mode in a simulation, thus allowing vehicles to follow a dynamically produced trajectory, choosing randomly when approaching a junction, repeating this behavior during the simulation.

The autonomous driving sensor suite provides several sensors that users can configure, e.g., LiDAR, cameras, radar, GNSS, IMU, among others. ROS integration is possible with CARLA as it offers a bridge that enables two-way communication between ROS and CARLA. CARLA also provides co-simulation support with SUMO (see Section IV-B), allowing to exploit the capabilities of both simulation tools within the same framework.

Since CARLA is based on the Unreal Engine, a 3D computer graphics engine, it has minimum hardware requirements, including a dedicated GPU of at least 8GB and 20GB of free disk space [28].

2) AWSIM

AWSIM [16] is an open-source simulator for self-driving vehicles based on the Unity game engine to emulate a realistic environment where vehicles equipped with sensors operate. The main features include vehicle dynamic models, sensor models, and environment configuration. To facilitate the integration with other tools, AWSIM provides communication based on ROS. AWSIM has a vehicle that can be configured to have multiple sensors (LiDAR, camera, IMU, and GNSS), and supports new vehicles added by users. Regarding the simulated environment, AWSIM provides a sample map of a scenario in Tokyo, Japan. The simulated environment of this realistic scenario is composed of the roads, buildings, traffic lights, vehicles, and pedestrians. New environments may also be added by users. To replicate the movement behavior of vehicles in the city, a random traffic simulator simulates city traffic considering all traffic rules. This traffic simulator allows for random selection of car models and the paths they follow. The communication is based on ROS, hence sensor data, status information, among others, can all be obtained from the topics where AWSIM publishes information.

3) DISCONTINUED OPEN-SOURCE SIMULATORS

In the following, we list some of the most prominent simulation tools that enable multi-sensor simulation of 3D realistic environment, which were discontinued.

The AirSim simulator [29], [46] is a simulation platform for developing and testing algorithms for autonomous vehicles. AirSim was developed as a platform for Artificial Intelligence (AI) research to experiment with deep learning and reinforcement learning algorithms for autonomous vehicles. In addition to the realistic environments and vehicle dynamics, AirSim provides multi-modal sensing, supporting sensors such as cameras, barometer, IMU, Global Positioning System (GPS), magnetometer, distance and LiDAR. Unfortunately, Microsoft Research announced the end of this project in 2022. They also announced a new simulator, called Microsoft Project AirSim, focusing on aerospace applications. To the best of our knowledge, this simulator is not available yet.

The LGSVL simulator [47], [48] is a free and open-source tool that aims to improve autonomous vehicle development with a high-fidelity simulation engine based on the Unity game engine. It provides end-to-end, full-stack simulation with support for various vehicle sensors, e.g., camera, radar, LiDAR, GPS, IMU, among others. To facilitate integration with other systems, the LGSVL simulator has a communication channel that enables to pass messages between the simulator and an autonomous driving stack, like Autoware or Baidu Apollo. Unfortunately, the development of this project was suspended on January 1st, 2022, with no plans for future improvements and bug fixes.

DeepDrive [49], [50] is an open-source simulator based on the Unreal Engine. It was developed to support the development of computer vision techniques based on AI for self-driving, hence it enables vehicles to be equipped with up to eight cameras with depth information. Supporting only cameras, makes this simulator more limited in comparison with other open-source ones. Besides, its documentation is not as detailed as CARLA's or AirSim's. Unfortunately, this tool was lastly updated in 2020.

4) COMMERCIAL SIMULATORS

There are many commercial simulators including Realtime Technologies SimCreator [55], AVSimulation SCANeR Studio [18], NVIDIA DRIVE Sim [56], ANSYS AVxcelerate

FIGURE 3. Examples of commercially available 3D simulation tools [17], [18], [51], [52], [53], [54].

Sensors [52], IPG Automotive CarMaker [17], dSPACE AURELION [51], rFpro [53], Cognata [54], CarSim [57], MORAI Sim Drive [58], Navigation Toolbox [59], and Automated Driving Toolbox [60]. These simulation tools (depicted in Fig. 3) focus on providing an easy-to-use app that offers a realistic simulation environment for users to test their systems. A feature comparison between open-source and commercial simulators as well as a discussion on the main differences between them is presented in Section VI.

TABLE 1. Supported sensors in vehicle simulators (top 5 are open-source simulators and the others are commercial ones).

Simulator	Cameras	GNSS/GPS	IMU	LiDAR	Radar	Ultras.	Others
AirSim [46]	✓$^{\beta}$	✓	✓	✓	✗	✗	✓$^{\theta}$
AWSIM [16]	✓*	✓	✓	✓	✗	✗	-
CARLA [15]	✓$^{\alpha}$	✓	✓	✓	✓	✗	✓$^{\eta}$
DeepDrive [49]	✓^{o}	✗	✗	✗	✗	✗	✗
LGSVL [47]	✓$^{\delta}$	✓	✓	✓	✓	✓	✓$^{\kappa}$
aiSim [61]	✓*	✓	✓	✓	✓	✓	-
ANSYS AVxcelerate [52]	✓*	✗	✗	✓	✓	✗	-
CarMaker [17]	✓*	✓	✗	✓	✓	✓	-
CarSim [57]	✓*	✗	✗	✗	✓	✗	✓$^{\nu}$
Cognata [54]	✓$^{\zeta}$	✗	✗	✓	✓	✗	-
dSPACE AURELION [51]	✓$^{\epsilon}$	✗	✗	✓	✓	✗	-
MATLAB Auto. Driving Toolbox [60]	✓$^{\gamma}$	✗	✗	✓	✓	✓	✓$^{\xi}$
MATLAB Nav. Toolbox [59]	✗	✓	✓	✗	✗	✗	✓$^{\iota}$
MORAI Sim Drive [58]	✓*	✓	✓	✓	✓	✗	-
NVIDIA DRIVE Sim [56]	✓*	✗	✓	✓	✓	✗	✗
rFpro [53]	✓*	✗	✗	✓	✓	✗	-
SCANeR Studio [18]	✓*	✓	✓	✓	✓	✓	✓$^{\lambda}$

$^{\alpha}$ RGB, depth, segmentation, optical flow, DVS; $^{\beta}$ RGB, depth, segmentation, optical flow, infrared; $^{\gamma}$ RGB, depth, segmentation, fisheye; $^{\delta}$ RGB, depth, segmentation; $^{\epsilon}$ RGB, depth, segmentation, optical flow, fisheye; $^{\zeta}$ RGB, infrared, long-distance; $^{\eta}$ collision, lane invasion, obstacle; $^{\theta}$ barometer, magnetometer, distance sensor; $^{\iota}$ wheel encoder, range finder, Inertial Navigation System (INS), altimeter; $^{\kappa}$ lane-line sensor, lane following sensor, comfort sensor; $^{\lambda}$ lighting sensor; $^{\nu}$ traffic signs sensor (camera), moving object detector; $^{\xi}$ vision detection generator (detects objects and lanes on images captured by a camera); ^{o} RGB, depth; * supported camera types are not specified.

5) SENSOR SUPPORT

Table 1 shows the supported sensors in analyzed simulators. CARLA, AirSim, and LGSVL are the open-source simulation tools with support for more types of sensors, some of which provide information for ADAS maneuvers, such as the collision, lane invasion and obstacle sensors in CARLA, and the lane-line, lane following and comfort sensors available in LGSVL simulator.

Each of the commercially available simulation tools usually focuses on different aspects of autonomous driving, hence not supporting some sensors. Despite that, SCANeR Studio is the one that supports more sensors, including a lighting sensor.

Cameras are crucial for autonomous driving and ADAS, thus all analyzed simulators support, at least, RGB cameras, with many supporting other types of cameras such as depth (Time-of-Flight (ToF)), segmentation, optical flow, fisheye, among others. In autonomous driving and ADAS, cameras are mostly used for object detection and recognition, lane detection and keeping, traffic sign recognition. In cooperative approaches, cameras are essential in loop closure techniques to identify overlapping areas between vehicles.

Regarding GPS and GNSS support, it is important to mention that most simulators provide simple models for receivers, just providing the receiver estimated positions, considering noise, which may be configured. These models are too simplistic and fail to emulate the satellite constellations, signal propagation, propagation effects, and pseudoranges. We performed a deeper analysis on GNSS simulators in Section IV-D, which are focused only on generating synthetic raw GNSS data. DeepDrive is the only open-source simulator that does not support GNSS, but there are numerous commercial simulators that have no support for this sensor, for instance, ANSYS AVxcelerate, CarSim, Cognata, dSPACE AURELION, MATLAB Automated Driving Toolbox, NVIDIA DRIVE Sim and rFpro.

IMU sensors are essential for tracking the movement and orientation of the vehicle. Typically, these sensors are composed of accelerometer, gyroscope, and sometimes a magnetometer, which output, the acceleration, angular velocity, and orientation based on Earth's magnetic field. Normally, IMUs are used for enhancing the tracking and positioning of the vehicle, as they allow to perform dead reckoning and enable sensor fusion with GNSS. Most open-source simulators support IMU sensor while the majority of commercial simulators lack support of this sensor. Hence, these simulators lack the functionality that allows developers to create sensor fusion techniques for better tracking based on IMU.

LiDAR sensors emit laser beams that reflect off nearby objects and return to the sensor, allowing precise measurement of distances. The resulting point cloud provides detailed spatial data that is crucial for autonomous driving, enabling the vehicle to detect obstacles, map its surroundings, and navigate safely. With the exception of DeepDrive, CarSim, and MATLAB Navigation Toolbox, all simulators support LiDAR sensors.

Radar, unlike lidar, uses radio frequency waves to measure the distance to nearby objects by detecting the time it takes for signals to reflect back. It provides data on the speed and position of objects, which are crucial for tasks like collision avoidance, adaptive cruise control, and detecting moving objects, even in adverse weather conditions. Most simulators support radar sensors, demonstrating its importance in perception of the surrounding environment. Only AirSim, AWSIM, DeepDrive and the MATLAB Navigation Toolbox lack support for radar sensors.

Ultrasound sensors emit high-frequency sound waves (low range) and measure the time it takes for them to return after bouncing off objects, thus allowing to estimate the distance. Ultrasound sensors are typically used for object detection at close-range, assisting in parking and other maneuvers at low speed, supporting precise maneuvers and enhanced perception in tight spaces. Most simulators lack the support for the ultrasound sensor, with the LGSVL, aiSim, CarMaker, MATLAB Automated Driving Toolbox, and SCANeR Studio being the simulators with support for ultrasound sensors.

In addition to the sensors previously mentioned, some simulators support additional sensors. These include the barometer, which measures air pressure and detects elevation changes; the altimeter, which measures altitude or elevation relative to a specific reference point; and the wheel encoder, which is essential for estimating the vehicle's trajectory through dead reckoning. Additionally, some simulators support meta-sensors that provide integrated data. These include collision sensors, lane-line sensors, lane-following sensors, lane-invasion sensors, comfort sensors, and obstacle sensors.

B. TRAFFIC AND MOBILITY

Simulating the mobility and traffic behavior of vehicles in cities can help in the definition of cooperative positioning scenarios, especially in early development stages where testing requires vehicles to move and behave in favorable conditions.

1) SUMO

SUMO [19], [62] stands for Simulator of Urban MObility and is an open-source, highly portable and continuous traffic simulation package designed to handle large networks. It was designed for simulating intermodal traffic systems including road vehicles, public transport, and pedestrians. Scenarios can be created and edited with a large set of tools provided by this simulator. SUMO allows the simulation of traffic demand, represented by individual vehicles, as they move through a given road network. It is microscopic, i.e., each vehicle is modelled explicitly, has its own route, and moves individually through the network. Integration with automated vehicles is possible in SUMO, as well as V2X communication with the possibility of integrating with communication network simulators like NS-3 or OMNeT++. SUMO can be enhanced with custom models and provides various APIs to remotely control the simulation. In the past years, SUMO has been used in many projects with different purposes, e.g.: evaluation of the performance of traffic lights; vehicle route choice, and evaluation of influences of autonomous route choice; V2X projects for providing realistic vehicle traces and for evaluating the application in an on-line loop with a network simulator; simulation of traffic effects of autonomous vehicles and platoons; simulation of cooperative positioning solutions.

2) VISSIM

Vissim [20] is a traffic and mobility simulation software developed by PVT. It allows to simulate multi-modal traffic of all road users, e.g., vehicles, pedestrians, cyclists, etc. Among other applications, this simulator allows to simulate the driving behavior of connected and autonomous vehicles, including interfaces to multi-sensor simulators, like CarMaker or CARLA. This enables a realistic and reactive surrounding traffic to virtual test drives, creating a continuous interaction with the system under test. Contrarily to SUMO, Vissim is closed-source and requires a paid license.

C. V2X COMMUNICATION

Simulating vehicular communications requires a specialized framework with the models considering the factors that affect communication. This includes defining protocols, network layers, and propagation models to capture real-world factors such as signal interference, multi-channel operation, and obstacles that affect communication. In this section, we present a selection of tools that enable V2X simulations, highlighting their features and compatibility with other simulators for vehicular network modelling.

1) OMNET++

OMNeT++ [21] is a modular and extensible simulation library and framework, primarily for building network simulators. Each module (component) is programmed in C++ and can be integrated into OMNeT++ architecture to enable new features. Numerous simulation models and model frameworks have been developed for OMNeT++ by the community in diverse areas, such as mobile ad-hoc networks, mesh networks, wireless sensor networks, vehicular networks, among others. The INET Framework [63] is an open-source model library for OMNeT++ which provides protocols, agents, and other models for communication networks. INET implements the OSI layers (physical, link-layer, network, transport, and application) and has models for the internet stack (IPv4, IPv6, Transmission Control Protocol (TCP), User Datagram Protocol (UDP), wired and wireless link-layer protocols (Ethernet, IEEE802.11), and supports modelling the physical environment (obstacles for radio propagation). Several other simulation frameworks take INET as a base and extend it into specific directions, such as vehicular networks.

Veins [64] is a simulation framework that uses INET as a basis and combines the OMNeT++ event-based network simulator with the SUMO traffic simulator using co-simulation. Veins enables inter-vehicular communication by providing models of IEEE 802.11p and IEEE 1609.4 DSRC/WAVE network layers, including multi-channel operation, noise, and interference effects, to make vehicular network simulations as realistic as possible. Veins works as follows. Road traffic simulation is performed by SUMO and network simulation is performed by OMNeT++. The network simulation includes physical layer modelling that makes it possible to apply accurate models for signal propagation, taking into consideration radio interference as well as shadowing by static or moving obstacles. SUMO and OMNeT++ are bi-directionally coupled so that simulations are performed online, allowing to model vehicular networks on road traffic considering the complex interactions between both domains of these simulators.

Artery [65] is a V2X simulation framework that supports ETSI ITS-G5. Single vehicles can be equipped with multiple ITS-G5 services through Artery's middleware. Basic services like Cooperative Awareness Messages (CAMs) and Decentralized Notification Messages (DENMs) are already included. Although it started as an extension of Veins, nowadays Artery can be run independently. Similarly to Veins, Artery also runs side-by-side with SUMO and OMNeT++. For the wireless communication model, Artery allows to choose between the IEEE 802.11 model from Veins or the INET framework, depending on the users' needs.

2) NETWORK SIMULATOR (NS-3)

The discrete-event network simulator NS-3 [22] is a free and open-source simulator for communication networks, supporting a wide range of technologies and protocols. It has a modular architecture, which encourages researchers to implement and share new modules with the community. This simulator has realistic network simulation by implementing protocol stacks, for instance, it includes TCP/IP full-stack implementation. NS-3 is commonly used in wireless networks simulation (Wi-Fi, LTE and 5G) and in Internet of Things (IoT) network simulation (LoRaWAN), thus being useful for mobile and wireless network research. It is also used to simulate vehicular communications, as shown by the modules described next.

MilliCar [66] is an NS-3 module for V2V mmWave networks, that implements the physical and medium access control (MAC) layers based on the 3GPP standard for next-generation vehicular systems.

MoReV2X [67] enables New Radio (NR) V2X communications, by complying with the 3GPP specifications, including the physical and MAC layers, as well as, the NR propagation channel. It implements a new set of realistic traffic models and it can also be easily interfaced with SUMO for the realistic simulation of vehicular mobility.

A **cellular V2X (C-V2X)** communication module for NS-3 was proposed in [68]. It is an open-source module based on LTE Device-to-Device communication model for the NS-3 network simulator. It implements C-V2X mode 4 and allows integration with vehicle traffic simulator SUMO.

D. RAW GNSS MEASUREMENTS

Some cooperative positioning approaches rely on vehicles exchanging raw GNSS data between them [34]. Other simulators analyzed in this paper may simulate the vehicles' behavior or a realistic environment for vehicles equipped with sensors, but they do not generate synthetic raw GNSS data, considering propagation effects and sources of errors. Raw GNSS data can be integrated into 3D realistic simulation tools, allowing to evaluate cooperative approaches based on pseudoranges sharing. Hence, in this section we present simulators for generating raw GNSS signals. Firstly, we detail hardware-based simulators, then we present two software-based simulators.

1) HARDWARE-BASED GNSS SIMULATORS

Hardware-based GNSS simulators, can be of two types namely, GNSS simulators or record & replay systems. The former is a system that simulates one or more constellations of satellites, by generating the signals that would be observed by a GNSS receiver in a specific location on Earth. The latter is a system that records GNSS data, which can then be reproduced for each receiver under test. These devices provide a RF (Radio Frequency) signal that has either been synthesized from a simulation model or has been captured from real live signals. GNSS simulation devices allow testing

GNSS, such as the Spirent GSS9000 [69] or the Rohde & Schwarz SMW200A [70], are expensive, costing over 100K €. The record & replay systems Spirent GSS6450 [71] and Labsat 3 [72] have a cost between 20K-30K €.

2) SOFTWARE-BASED GNSS SIMULATORS

The **MATLAB Satellite Communications Toolbox** [73] allows to simulate satellite communications systems and links. Among the features provided, this toolbox includes RF propagation and channel models that consider the atmospheric loss in simulated signals. When combined with the Navigation Toolbox [59], it allows to estimate the GNSS receiver position with simulated satellite constellations. The Satellite Communications Toolbox simulates the satellite positions over time via a RINEX file, then the Navigation Toolbox provides functions to simulate the satellite signal processing of the receiver, calculate the pseudoranges and estimate the receiver position. Since the release `R2023a` the Navigation Toolbox also includes a GNSS raw measurements generator [74], which allows to simulate GNSS receiver measurements, i.e., the pseudoranges based on sensor time and satellite orbits. This toolbox can be used for simulating GNSS multipath effects in urban canyon environments [75], generating raw GNSS pseudoranges affected by multipath caused by the urban environment.

The **SatNav Toolbox for MATLAB** [76] developed by GPSoft allows to simulate the satellites, the propagation environment, the receiver measurements, and includes the data processing for the GNSS measurements. This toolbox simulates the satellites and receivers, as well as the propagation channels. In addition, error sources such as thermal noise, multipath, atmospheric delays and Selective Availability are modelled as an integral part of pseudorange and integrated Doppler (carrier-phase) emulation. With this toolbox, users can simulate constellations like Galileo and perform real-data processing via a RINEX2 file. The SatNav Toolbox also provides several functions that allow to: produce skyplots of satellite positions; estimate positions based on real pseudorange measurements; compute and analyze the effects of dilution-of-precision (DOP); compute the parity vector which is a test statistic used for fault detection and isolation (FDI).

V. USING SIMULATION TOOLS FOR COOPERATIVE POSITIONING

Although there are simulation tools that meet some of the requirements listed in Section III-E, there is no simulation tool that is able to meet all of these requirements. To overcome this, researchers often try to integrate different simulation tools for the development of cooperative positioning solutions [77], [78], [79], since combining multiple tools allows to benefit from features that each tool provides.

CARLA was used in [77] to evaluate a cooperative positioning approach that estimates inter-vehicle poses via vision and LiDAR sensors. In this paper, a cooperative awareness approach is proposed, where vehicles interact with each other exchanging information from cameras and LiDARs to determine their relative positions, then using this information with two localization schemes based on Gradient Descent and Extended Kalman Filter to mitigate GPS errors. This approach was assessed with extensive simulations using CARLA.

GNSS RUMS [78], stands for Realistic Urban Multiagent Simulator, is a simulator for collaborative positioning research that provides synthetic GNSS pseudorange measurements. It employs ray-tracing techniques considering the 3D model of the space to categorize signals as direct, reflected, diffracted and multipath, to then simulate pseudorange, carrier-to-noise ratio and Doppler shift measurements. This simulator is integrated with SUMO to simulate GNSS measurements from multiple agents with distinct transportation behaviors. Validation of this simulator was performed by comparing synthetic measurements with real measurements. Conducted experiments with different positioning algorithms showed that this simulator is sophisticated enough for testing collaborative positioning approaches. Unfortunately, this simulator is not available to use.

Soatti et al. [79] proposed an implicit cooperative positioning framework in which vehicles jointly localize non-cooperative physical features (such as people, traffic lights, or inactive cars) in the surrounding areas, and use them as common noisy reference points to refine their location estimates. This method was validated in a real urban scenario using the SUMO traffic simulator which used real maps to generate synthetic traces of vehicles and pedestrians.

Although this paper focuses on simulation tools for cooperative positioning, several works have also combined multiple simulators for cooperative driving applications, such as perception and cooperative maneuvers.

OpenCDA [80] is a framework for developing and testing Cooperative Driving Automation (CDA) systems. Using co-simulation, it combines CARLA and SUMO to bring realistic environment simulation with mobility models, from these simulators. OpenCDA also provides a platform for automated driving and cooperative driving with modules for perception, localization, planning control and V2X communication.

Malinverno et al. [81] propose an open-source simulation framework for NS-3 integrating communication stacks, including V2X protocols with the SUMO simulator, which manages the node mobility in the simulation. Carletti et al. [45] extends [81] by integrating CARLA (high-fidelity sensor and vehicle physics), OpenCDA data fusion and control automation, with NS-3's communication models, providing a framework for vehicular cooperative perception with different communication technologies.

Jooriah et al. [44] introduce a co-simulation framework for autonomous driving scenarios combining CARLA and Artery via ROS network. Artery enables CARLA vehicles to communicate using V2X protocols. The goals of their paper are to establish a connected ecosystem with V2X communication infrastructure for autonomous driving and ADAS development.

Aramrattana et al. [82] presented a framework that combines driving-, traffic-, and network-simulators. More specifically, this framework integrated the driving simulator VTI [83] with an integrated traffic-network simulator, Plexe [84] which is a cooperative driving simulator extending SUMO and Veins (see Sections IV-B and IV-C) permitting the realistic simulation of platooning systems. Simulation results were provided for several platooning scenarios. This simulation framework provides great opportunities by combining the three types of simulators because the driving simulator allows studies from the driver's perspective, the network simulator addresses communications-related issues, and the traffic simulator allows an analysis of the collaborative approaches on traffic systems.

As mentioned above, Plexe [84] is an open-source extension to Veins, providing a simulation environment for realistic cooperative driving experiments. It considers vehicle physics and mechanics, communication and networking challenges, and Inter-Vehicle Communication (IVC) protocol stacks. Plexe is easily extensible and implements protocols to support cooperative driving applications, such as platooning. To achieve that, vehicular communication stack is enabled by Veins extending the OMNeT++ simulator, whilst the realistic node mobility for cooperative is modelled based on the SUMO simulator.

VI. SUMMARY AND DISCUSSION

Table 2 summarizes the analyzed simulation tools that enable cooperative positioning, considering the type of software (open or closed source), the platform compatibility (operating system), the engine, and the cost. Also, it shows a comparison of supported features, namely: **Vehicle Sensors** – support for vehicle sensors like cameras, LiDAR, or radar, etc. (see Table 1 for further details); **3D Env. & Physics** – 3D environment representation with realistic visuals and physics; **Traffic & Mobility** – traffic and mobility generation, regarding the vehicle's movement and behavior; **V2X** – Vehicle-to-Everything (V2X) communication protocols; **Raw GNSS** – generation of raw GNSS measurements, considering satellites' orbits, atmosphere effects, propagation effects, such as, multipath, and receiver noise; **SIL** – Software-In-the-Loop (SIL) support; **HIL** – Hardware-In-the-Loop (HIL) support; **Co-sim.** – co-simulation, i.e., integration with other simulators; **ROS** – communication interface for the Robot Operating System (ROS).

Realistic multi-sensor simulators based on game engines are used mostly for sensor development, validation and testing of autonomous driving and advanced driver assistance system features. Most of these simulators allow integration with autonomous driving stacks such as Apollo or Autoware, and support SIL or HIL. These simulators are also suitable for representing cooperative positioning scenarios, where vehicles equipped with multiple sensors exchange information to improve their position estimates. Due to the visually and physically realistic environments these simulators create, they have demanding hardware requirements. A dedicated graphics card and a powerful CPU are necessary to run the game engines effectively.

When considering open-source and free simulators, CARLA seems to be the best multi-sensor simulator for several reasons: 1) it has the necessary features to simulate a cooperative positioning scenario, namely, it is realistic, supports multiple vehicles, supports different sensors (IMU, GNSS, LiDAR, radar, cameras, etc.), allows to perform tests under different weather conditions; 2) it is still under development and being updated; 3) it has up to date and detailed documentation; 4) it has an online forum for sharing tips and solving issues. In comparison with AirSim and LGSVL, CARLA has similar features, but AirSim and LGSVL development has stopped, therefore they are not a good alternative to consider nowadays. In comparison with AWSIM, CARLA offers more resources (assets), includes more ego-vehicles that can be equipped with sensors, and has more maps than AWSIM, so it is a better alternative. Although CARLA and AWSIM have traffic managers to automatically manage the creation of vehicles and the way they move in the simulation, these models are simplistic in comparison with traffic and mobility simulators like SUMO and Vissim. To enable more realistic traffic models and algorithms for vehicle movement and interaction, some simulators support co-simulation with SUMO, e.g., CARLA supports co-simulation with SUMO.

The vehicle multi-sensor commercial simulators are paid tools and closed-source, hence they are not as easily extensible as open-source tools. Despite that, tools like NVIDIA DRIVE Sim or dSPACE AURELION have open Application Programming Interfaces (APIs) and support custom modules that are easily integrated into the simulation framework.

Simulating communications between vehicles and other road actors in cooperative environments requires several frameworks, considering a traffic and mobility simulator (e.g., SUMO), an event-based network simulator (e.g., OMNeT++), and the vehicular communication framework (e.g., Veins or Artery) that considers the propagation model, affected by interference and other effects. Researchers have developed tools that integrate traffic and mobility with vehicular communications frameworks to support cooperative driving scenarios [84].

Despite that some simulators provide GPS/GNSS sensor models, neither the open-source nor the commercially available multi-sensor simulators, like CARLA, AVSimulation SCANeR Studio, NVIDIA DRIVE Sim, etc., are capable of generating synthetic GNSS raw data which is necessary for cooperative positioning techniques based on vehicles exchanging pseudoranges. That can be achieved using software or hardware-based simulators. Software simulators are usually lower-cost than hardware-based ones, hence, the GPSoft SatNav Toolbox or the MATLAB Satellite Communications Toolbox are good options in terms of cost

TABLE 2. Comparison between simulation tools for cooperative positioning.

Simulator	Type	Software	Compatibility	Engine	Cost	Vehicle Sensors	3D Env. & Physics	Traffic & Mobility	V2X	Raw GNSS	SIL	HIL	Co-sim.	ROS
AirSim [46]	VM	Open source (C++, Python)	Windows, Linux, macOS	Unreal	Free	✓	✓	✗	✗	✗	✓	✓	✗	✓
AWSIM [16]	VM	Open source (C#)	Linux	Unity	Free	✓	✓	✓	✓$^{\iota}$	✗	✗	✗	✗	✓
CARLA [15]	VM	Open source (C++, Python)	Windows, Linux	Unreal	Free	✓	✓	✓	✗	✗	✗	✗	✓$^{\kappa}$	✓
DeepDrive [49]	VM	Open source (C++, Python)	Windows, Linux	Unreal	Free	✓	✓	✗	✗	✗	✗	✗	✗	✗
LGSVL [47]	VM	Open source (C#)	Windows, Linux	Unity	Free	✓	✓	✗	✓	✗	✓	✓	✓$^{\mu}$	✓
aiSim [61]	VM	Closed source	Windows, Linux	Unreal	Paid	✓	✓	–	✗	✗	✓	✓	✓$^{\chi}$	✓
ANSYS AVxcelerate [52]	VM	Closed source	Windows, Linux	–	Paid	✓	✓	–	✗	✗	✓	✓	✓$^{\nu}$	–
CarMaker [17]	VM	Closed source (C, C++)	Windows, Linux	MovieNX 3D, Unigine	Paid	✓	✓	✗	✗	✗	✓	✓	–	–
CarSim [57]	VM	Closed source	Windows, Linux	–	Paid	✓	✓	–	✗	✗	✓	✓	✓^{o}	✗
Cognata [54]	VM	Closed source	Microsoft Azure	Unity	Paid$^{\gamma}$	✓	✓	✓	✗	✗	–	✓	✓$^{\rho}$	✓
dSPACE AURELION [51]	VM	Closed source	Windows, Linux	Unreal	Paid	✓	✓	–	✗	✗	✓	✓	–	✗
MATLAB Auto. Driving Toolbox [60]	VM	Closed source	Windows, Linux, macOS	RoadRunner, Unreal$^{\eta}$	Paid$^{\beta}$	✓	✓	✗	✗	✗	✓	✓	✓$^{\lambda}$	✓$^{\phi}$
MATLAB Nav. Toolbox [59]	VM	Closed source	Windows, Linux, macOS	–	Paid$^{\alpha}$	✓	✗	✗	✗	✓$^{\theta}$	✗	✗	✗	✓$^{\phi}$
MORAI Sim Drive [58]	VM	Closed source	Windows, Linux, AWS	Unity	Paid$^{\delta}$	✓	✓	✓	–	✗	✗	✓	–	–
NVIDIA DRIVE Sim [56]	VM	Closed source (C++, Python)	–	Nvidia Omniverse	–	✓	✓	✓	–	–	–	–	–	–
rFpro [53]	VM	Closed source (C++)	Windows, Linux	Proprietary	Paid	✓	✓	✓	✗	✗	✓	✓	✓$^{\pi}$	✗
SCANeR Studio [18]	VM	Open software (C++)	Windows, Linux	Unreal	Paid	✓	✓	–	✗	✗	✓	✓	–	–
SUMO [19]	TM	Open source (C++, Python)	Windows, Linux, macOS	–	Free	✗	✗	✓	✗	✗	✗	✗	✓$^{\sigma}$	✗
Vissim [20]	TM	Closed source (C++)	Windows	–	Paid	✗	✗	✓	✗	✗	✗	✗	✗	✗
NS-3 [22]	DE	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\omega}$	✗
OMNeT++ [21]	DE	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\tau}$	✗
Artery [65]	VC	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\upsilon}$	✗
Veins [64]	VC	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\upsilon}$	✗
MilliCar [66]	VC	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\psi}$	✗
MoReV2X [67]	VC	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\psi}$	✗
Cellular V2X [68]	VC	Open source (C++)	Windows, Linux, macOS	–	Free	✗	✗	✗	✓	✗	✗	✗	✓$^{\psi}$	✗
GPSoft SatNav Toolbox 3.0 [76]	GNSS	Closed source	MATLAB	–	Paid$^{\zeta}$	✗	✗	✗	✗	✓	✗	✗	✗	✗
MATLAB Sat. Comm. Toolbox [73]	GNSS	Closed source	Windows, Linux, macOS	–	Paid$^{\epsilon}$	✗	✗	✗	✓	✗	✗	✗	✗	✗

VM – Vehicle Multi-sensors in 3D Realistic Environment; TM – Traffic and Mobility; DE – Discrete Event; VC – V2X Communications; GNSS – Raw GNSS measurements; "–" – Not applicable/Not Available; $^{\alpha}$ requires MATLAB license plus the toolbox license (2250€perpetual or 900€annually); $^{\beta}$ requires MATLAB license plus the toolbox license (price unavailable); $^{\gamma}$ Cloud-based version (price unavailable); $^{\delta}$ Cloud-based version from 0.752$/h; $^{\epsilon}$ requires MATLAB license plus the toolbox license to use (10750€perpetual or 4300€annually); $^{\eta}$ only available on Windows; $^{\theta}$ requires Automated Driving Toolbox which has GNSS sensor model for vehicles in order to generate raw GNSS measurements in a driving scenario; $^{\iota}$ supports only Vehicle-to-Infrastructure communications; $^{\kappa}$ SUMO, VISSIM; $^{\lambda}$ Unreal Engine; $^{\mu}$ Baidu Apollo, Autoware; $^{\nu}$ CarMaker, CARLA; ^{o} dSPACE, Unreal Engine, MATLAB; $^{\pi}$ SUMO, CarMaker, VTD, MATLAB; $^{\rho}$ SUMO, MATLAB; $^{\sigma}$ NS-3, OMNeT++; $^{\omega}$ SUMO, MilliCar, MoReV2X, C-V2X; $^{\tau}$ Veins, Artery; $^{\upsilon}$ OMNeT++, SUMO; $^{\psi}$ NS-3, SUMO; $^{\phi}$ integration enabled by ROS Toolbox. $^{\chi}$ Autoware, Unreal Engine, Foretellix Foretify, MATLAB, dSPACE.

and features, since they allow generating raw pseudoranges from different satellite constellations.

Regarding the cost of these tools, it is relevant to mention that most commercial tools require a paid license, which can vary depending on the type of license (single device, or multi device) and version (local install or cloud-based version). Tools like MATLAB's toolboxes require the MATLAB license, plus the toolbox license, which can be expensive, in the order of several thousands of euros.

VII. CONCLUSION AND FUTURE WORK

In this paper, we reviewed the tools necessary to simulate a cooperative positioning scenario. We started with an overview of cooperative positioning approaches and presented the respective simulation requirements. Then we detailed the available simulation tools that enable the simulation of different components of a cooperative scenario. From this study, we found that no standalone tool can encompass all aspects of simulating a cooperative positioning scenario, including realistic vehicle movement, sensor integration, and communication between vehicles. Due to the complexity of simulating the entire cooperative positioning pipeline, combining multiple simulation tools is the most effective approach. By leveraging existing simulation tools, researchers can allocate more efforts towards developing innovative solutions.

There are numerous multi-sensor simulators to choose, from open-source simulators like CARLA or AWSIM to commercially available simulators like CarSim, Cognata, dSPACE AURELION, MATLAB Toolboxes, among others. The feature set these simulators provide, including a high-fidelity 3D environment, realistic physics, multiple sensor support, as well as, APIs support and co-simulation, make them suitable not only for autonomous driving, but also for cooperative positioning scenarios, as they allow to simulate multiple vehicles with positioning and perception sensors.

Although multi-sensor simulators have integrated traffic managers to handle the traffic and behavior of vehicles, these models are usually simplistic. Hence, to properly simulate cooperative scenarios, multi-sensor simulators that support co-simulation can integrate traffic and mobility simulators like SUMO, which provide more accurate and realistic traffic and mobility models, and can be integrated into multi-sensor simulators that support co-simulation.

The ability to exchange data between road participants is crucial for cooperative positioning. Simulation of vehicular communications is provided by simulators like OMNeT++ or NS-3 for which researchers have implemented communications with packages like Veins or MoReV2X. These packages consider the networking protocols as well as the propagation channel which is affected by propagation effects.

GNSS-based cooperative positioning techniques rely on the exchange of raw GNSS measurements between vehicles. Despite the existence of both hardware and software-based GNSS simulators, there is no raw GNSS measurements simulator that integrates with the other simulators that allow to simulate a cooperative scenario.

In this study, we also reviewed several papers utilizing simulation tools for the development of cooperative positioning solutions, and cooperative driving applications. Most works combine different simulators to leverage the capabilities of each one. For instance, GNSS RUMS generates synthetic GNSS pseudoranges and is integrated with SUMO for realistic traffic models. Similarly, CARLA and SUMO have been used together to simulate both high-fidelity vehicle dynamics, sensors' data, and traffic models to enable the testing of cooperative perception systems.

Depending on the development stage, certain components of the simulation pipeline, such as mobility and networking, may be simplified or omitted initially. This allows for streamlined development, e.g., assuming perfect communication without losses. Although this approach may lead to overoptimistic results, it facilitates the initial stages of system development. Nevertheless, to comprehensively evaluate a cooperative positioning system in simulation, it is crucial to consider the complete pipeline as mentioned above. This ensures a more realistic and thorough assessment of the system's performance.

In the future, we aim to develop a cooperative positioning simulation framework to support GNSS reference-based cooperative positioning scenarios, using CARLA combined with the SatNav Toolbox. CARLA was selected for its robust support in simulating cooperative scenarios. Built on Unreal Engine, it provides high-fidelity representation of vehicles and the surrounding environment with realistic physics. It offers multi-sensor simulation, supporting GNSS, LiDAR, radar, cameras, among others, facilitating not only cooperative positioning but also autonomous driving testing. Additionally, CARLA includes predefined mobility models for vehicle dynamics, traffic scenarios, and supports co-simulation with tools like SUMO. While some open-source simulators have been discontinued, CARLA remains actively developed with ongoing updates and GitHub community support. Although it lacks certain commercial features like HIL or SIL, its open-source nature and accessibility make it a preferred choice for research. For raw GNSS measurements, the SatNav Toolbox offers an effective cost-benefit balance as a software-based solution, incorporating essential error sources like atmospheric effects and multipath, at a fraction of hardware-based simulator costs. To ensure V2X communication between vehicles, allowing them to exchange raw pseudoranges, co-simulation is required between CARLA and SUMO. Vehicular communications are enabled by frameworks like OMNeT++, which connects with SUMO and includes various modules that implement V2X communications protocols.

REFERENCES

[1] E. D. Kaplan and C. J. Hegarty, *Understanding GPS/GNSS: Principles and Applications*, 3rd ed. Artech House, 2017. [Online]. Available: https://ieeexplore.ieee.org/document/9100468

[2] C. Pendão, A. G. Ferreira, A. Moreira, C. Martins, and H. Silva, "Challenges in characterization of GNSS precise positioning systems for automotive," in *Proc. Int. Conf. Localization (ICL-GNSS)*, 2020, pp. 1–13. [Online]. Available: http://ceur-ws.org/Vol-2626/paper4.pdf

[3] X. Li, Z. Qin, Z. Shen, X. Li, Y. Zhou, and B. Song, "A high-precision vehicle navigation system based on tightly coupled PPP-RTK/INS/Odometer integration," *IEEE Trans. Intell. Transp. Syst.*, vol. 24, no. 2, pp. 1855–1866, Feb. 2023.

[4] N. Joubert, T. G. R. Reid, and F. Noble, "Developments in modern GNSS and its impact on autonomous vehicle architectures," in *Proc. IEEE Intell. Vehicles Symp. (IV)*, Oct. 2020, pp. 2029–2036.

[5] F. D. P. Müller, "Cooperative relative positioning for vehicular environments," Ph.D. dissertation, Univ. Passau, Passau, Germany, 2015. [Online]. Available: https://elib.dlr.de/121786/

[6] F. D. P. Müller, "Survey on ranging sensors and cooperative techniques for relative positioning of vehicles," *Sensors*, vol. 17, no. 2, p. 271, Jan. 2017. [Online]. Available: http://www.mdpi.com/1424-8220/17/2/271

[7] N. Alam, A. Kealy, and A. G. Dempster, "Cooperative inertial navigation for GNSS-challenged vehicular environments," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1370–1379, Sep. 2013. [Online]. Available: https://ieeexplore.ieee.org/document/6517506

[8] S. Malik, M. A. Khan, and H. El-Sayed, "Collaborative autonomous driving—A survey of solution approaches and future challenges," *Sensors*, vol. 21, no. 11, pp. 1–24, 2021.

[9] C. Qian, H. Zhang, W. Li, J. Tang, H. Liu, and B. Li, "Cooperative GNSS-RTK ambiguity resolution with GNSS, INS, and LiDAR data for connected vehicles," *Remote Sens.*, vol. 12, no. 6, p. 949, Mar. 2020. [Online]. Available: https://www.mdpi.com/2072-4292/12/6/949/htm

[10] E. Ghanem, K. O'Keefe, and R. Klukas, "Testing vehicle-to-vehicle relative position and attitude estimation using multiple UWB ranging," in *Proc. IEEE 92nd Veh. Technol. Conf.*, Nov. 2020, pp. 1–5.

[11] W. Wen, X. Bai, G. Zhang, S. Chen, F. Yuan, and L.-T. Hsu, "Multi-agent collaborative GNSS/Camera/INS integration aided by inter-ranging for vehicular navigation in urban areas," *IEEE Access*, vol. 8, pp. 124323–124338, 2020.

[12] (2024). *Driving Simulator | VI-Grade*. Accessed: Oct. 30, 2024. [Online]. Available: https://www.vi-grade.com/

[13] (2024). *Simdriver—Driving Simulator*. Accessed: Oct. 30, 2024. [Online]. Available: https://www.simdriver.pt/

[14] (2024). *Stisim Drive: Driver Training*. Accessed: Oct. 30, 2024. [Online]. Available: https://stisimdrive.com/driver-training/

[15] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Proc. 1st Annu. Conf. Robot Learn.*, S. Levine, V. Vanhoucke, and K. Goldberg, Eds., Nov. 2017. [Online]. Available: https://proceedings.mlr.press/v78/dosovitskiy17a.html

[16] (2023). *TIER4/AWSIM: Open Source Simulator for Self-Driving Vehicles*. Accessed: Sep. 6, 2024. [Online]. Available: https://github.com/tier4/AWSIM

[17] (2023). *CarMaker | IPG Automotive*. Accessed: Sep. 6, 2024. [Online]. Available: https://ipg-automotive.com/en/products-solutions/software/carmaker/

[18] (2023). *SCANeR studio—AVSimulation*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.avsimulation.com/scaner-studio/

[19] (2023). *SUMO—Simulation of Urban MObility*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.eclipse.org/sumo/

[20] P. P. T. V. GmbH. (2024). *Traffic Simulation Software | PTV Vissim*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.ptvgroup.com/en/products/ptv-vissim

[21] (2023). *OMNeT++ Discrete Event Simulator*. Accessed: Sep. 6, 2024. [Online]. Available: https://omnetpp.org/

[22] (2024). *ns-3 | A Discrete-Event Network Simulator for Internet Systems*. Accessed: Sep. 18, 2024. [Online]. Available: https://www.nsnam.org

[23] J. Craighead, R. Murphy, J. Burke, and B. Goldiez, "A survey of commercial & open source unmanned vehicle simulators," in *Proc. IEEE Int. Conf. Robot. Autom.*, Apr. 2007, pp. 852–857.

[24] D. Cook, A. Vardy, and R. Lewis, "A survey of AUV and robot simulators for multi-vehicle operations," in *Proc. IEEE/OES Auto. Underwater Vehicles (AUV)*, Oct. 2014, pp. 1–8.

[25] F. Rosique, P. J. Navarro, C. Fernández, and A. Padilla, "A systematic review of perception system and simulators for autonomous vehicles research," *Sensors*, vol. 19, no. 3, p. 648, Feb. 2019. [Online]. Available: https://www.mdpi.com/1424-8220/19/3/648/

[26] P. Kaur, S. Taghavi, Z. Tian, and W. Shi, "A survey on simulators for testing self-driving cars," in *Proc. 4th Int. Conf. Connected Auto. Driving (MetroCAD)*, Jan. 2021, pp. 62–70.

[27] Y. Li, W. Yuan, S. Zhang, W. Yan, Q. Shen, C. Wang, and M. Yang, "Choose your simulator wisely: A review on open-source simulators for autonomous driving," *IEEE Trans. Intell. Vehicles*, vol. 9, no. 5, pp. 4861–4876, May 2024, doi: 10.1109/TIV.2024.3374044.

[28] (2023). *CARLA Simulator*. Accessed: Sep. 6, 2024. [Online]. Available: https://carla.org/

[29] (2021). *AirSim Documentation*. Accessed: Sep. 6, 2024. [Online]. Available: https://microsoft.github.io/AirSim/

[30] F. J. Martinez, C. K. Toh, J.-C. Cano, C. T. Calafate, and P. Manzoni, "A survey and comparative study of simulators for vehicular ad hoc networks (VANETs)," *Wireless Commun. Mobile Comput.*, vol. 11, no. 7, pp. 813–828, 2011.

[31] J. S. Weber, M. Neves, and T. Ferreto, "VANET simulators: An updated review," *J. Brazilian Comput. Soc.*, vol. 27, no. 1, p. 8, Dec. 2021, doi: 10.1186/s13173-021-00113-x.

[32] S. Babu and A. R. Kumar P, "A comprehensive survey on simulators, emulators, and testbeds for VANETs," *Int. J. Commun. Syst.*, vol. 35, no. 8, p. e5123, 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/dac.5123

[33] S. Abdul Razak, S. Yogarayan, A. Azman, M. Abdullah, A. M. Amin, and M. Salleh, "Simulation framework for connected vehicles: A scoping review [version 2; peer review: 2 approved]," *F1000Research*, vol. 10, no. 1265, pp. 1–24, 2023.

[34] R. Jin, G. Zhang, L.-T. Hsu, and Y. Hu, "A survey on cooperative positioning using GNSS measurements," *IEEE Trans. Intell. Vehicles*, early access, May 7, 2024, doi: 10.1109/TIV.2024.3397879.

[35] R. Parker and S. Valaee, "Vehicle localization in vehicular networks," in *Proc. IEEE Veh. Technol. Conf.*, Sep. 2006, pp. 1–5. [Online]. Available: http://ieeexplore.ieee.org/document/4109822/

[36] J. Roth, T. Schaich, and G. F. Trommer, "Cooperative GNSS-based method for vehicle positioning," *Gyroscopy Navigat.*, vol. 3, no. 4, pp. 245–254, Oct. 2012. [Online]. Available: https://link.springer.com/article/10.1134/S2075108712040104

[37] S. Severi, h. Wymeersch, J. Härri, M. Ulmschneider, B. Denis, and M. Bartels, "Beyond GNSS: Highly accurate localization for cooperative-intelligent transport systems," in *Proc. IEEE Wireless Commun. Netw. Conf. (WCNC)*, Apr. 2018, pp. 1–6. [Online]. Available: https://ieeexplore.ieee.org/document/8377457/

[38] K. lassoued, P. Bonnifait, and I. Fantoni, "Cooperative localization with reliable confidence domains between vehicles sharing GNSS pseudoranges errors with no base station," *IEEE Intell. Transp. Syst. Mag.*, vol. 9, no. 1, pp. 22–34, Spring. 2017. [Online]. Available: https://ieeexplore.ieee.org/document/7823107/

[39] N. Alam, A. T. Balaei, and A. G. Dempster, "Relative positioning enhancement in VANETs: A tight integration approach," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 1, pp. 47–55, Mar. 2013. [Online]. Available: http://ieeexplore.ieee.org/document/6236174/

[40] V. Lepetit, F. Moreno-Noguer, and P. Fua, "EPnP: An accurate O(n) solution to the PnP problem," *Int. J. Comput. Vis.*, vol. 81, no. 2, pp. 155–166, Feb. 2009, doi: 10.1007/s11263-008-0152-6.

[41] L. Yin, Q. Ni, and Z. Deng, "A GNSS/5G integrated positioning methodology in D2D communication networks," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 2, pp. 351–362, Feb. 2018. [Online]. Available: http://ieeexplore.ieee.org/document/8287016/

[42] W. Zhang, Y. Yang, A. Zeng, and Y. Xu, "A GNSS/5G integrated three-dimensional positioning scheme based on D2D communication," *Remote Sens.*, vol. 14, no. 6, p. 1517, Mar. 2022. [Online]. Available: https://www.mdpi.com/2072-4292/14/6/1517/htm

[43] Y. Li, D. Ma, Z. An, Z. Wang, Y. Zhong, S. Chen, and C. Feng, "V2X-sim: Multi-agent collaborative perception dataset and benchmark for autonomous driving," *IEEE Robot. Autom. Lett.*, vol. 7, no. 4, pp. 10914–10921, Oct. 2022. [Online]. Available: https://ieeexplore.ieee.org/document/9835036/

[44] M. Jooriah, D. Datsenko, J. Almeida, A. Sousa, J. Silva, and J. Ferreira, "A co-simulation platform for V2X-based cooperative driving automation systems," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, May 2024, pp. 227–230. [Online]. Available: https://ieeexplore.ieee.org/document/10575990/

[45] C. M. R. Carletti, C. Casetti, J. Härri, and F. Risso, "MS-VAN3T-CARLA: An open-source co-simulation framework for cooperative perception evaluation," in *Proc. 19th Wireless Demand Netw. Syst. Services Conf. (WONS)*, Jan. 2024, pp. 93–96. [Online]. Available: https://ieeexplore.ieee.org/document/10449533/

[46] S. Shah, D. Dey, C. Lovett, and A. Kapoor, *AirSim: High-Fidelity Visual and Physical Simulation for Autonomous Vehicles* (Springer Proceedings in Advanced Robotics), vol. 5. Cham, Switzerland: Springer, May 2017, pp. 621–635.

[47] (2024). *LGSVL Simulator Github*. Accessed: Sep. 6, 2024. [Online]. Available: https://github.com/lgsvl/simulator

[48] G. Rong, B. H. Shin, H. Tabatabaee, Q. Lu, S. Lemke, M. Možeiko, E. Boise, G. Uhm, M. Gerow, S. Mehta, E. Agafonov, T. H. Kim, E. Sterner, K. Ushiroda, M. Reyes, D. Zelenkovsky, and S. Kim, "LGSVL simulator: A high fidelity simulator for autonomous driving," in *Proc. IEEE 23rd Int. Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2020, pp. 1–6.

[49] (2024). *DeepDrive*. Accessed: Sep. 6, 2024. [Online]. Available: https://deepdrive.io

[50] (2024). *DeepDrive Github*. Accessed: Sep. 6, 2024. [Online]. Available: https://github.com/deepdrive/deepdrive

[51] (2023). *dSPACE—AURELION—dSPACE*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.dspace.com/en/pub/home/products/sw/experimentandvisualization/aurelionsensor-realisticsim.cfm

[52] (2023). *Ansys AVxcelerate Sensors | AV Sensor Simulation Software*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.ansys.com/products/av-simulation/ansys-avxcelerate-sensors

[53] (2023). *Simulation Software | rFpro*. Accessed: Sep. 6, 2024. [Online]. Available: https://rfpro.com/simulation-software/

[54] (2023). *Cognata | Autonomous and ADAS Vehicles Simulation*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.cognata.com/autonomous-vehicles/

[55] (2023). *Real Time Simulation & Modeling System | SimCreator*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.faac.com/realtime-technologies/products/simcreator/

[56] (2023). *NVIDIA DRIVE Sim*. Accessed: Sep. 6, 2024. [Online]. Available: https://developer.nvidia.com/drive/drive-sim

[57] (2023). *CarSim Mechanical Simulation*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.carsim.com/products/carsim/

[58] (2023). *Drive | MORAI Inc.* Accessed: Sep. 6, 2024. [Online]. Available: https://www.morai.ai/drive

[59] (2023). *MathWorks Navigation Toolbox*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.mathworks.com/products/navigation.html

[60] *Automated Driving Toolbox—MATLAB*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.mathworks.com/products/automated-driving.html

[61] (2024). *AISIM—Aimotive*. Accessed: Sep. 6, 2024. [Online]. Available: https://aimotive.com/aisim

[62] P. A. Lopez, M. Behrisch, L. Bieker-Walz, J. Erdmann, Y.-P. Flötteröd, R. Hilbrich, L. Lücken, J. Rummel, P. Wagner, and E. Wiessner, "Microscopic traffic simulation using SUMO," in *Proc. 21st Int. Conf. Intell. Transp. Syst. (ITSC)*, Nov. 2018, pp. 2575–2582. [Online]. Available: https://ieeexplore.ieee.org/document/8569938/

[63] (2023). *INET Framework for OMNeT++*. Accessed: Sep. 6, 2024. [Online]. Available: https://inet.omnetpp.org/

[64] C. Sommer, R. German, and F. Dressler, "Bidirectionally coupled network and road traffic simulation for improved IVC analysis," *IEEE Trans. Mobile Comput.*, vol. 10, no. 1, pp. 3–15, Jan. 2011.

[65] (2023). *Artery V2X Simulation Framework*. Accessed: Sep. 6, 2024. [Online]. Available: http://artery.v2x-research.eu/

[66] M. Drago, T. Zugno, M. Polese, M. Giordani, and M. Zorzi, "MilliCar: An ns-3 module for mmWave NR V2X networks," in *Proc. ACM Int. Conf.*, Feb. 2020, pp. 9–16.

[67] L. Lusvarghi and M. L. Merani, "MoReV2X—A new radio vehicular communication module for ns-3," in *Proc. IEEE 94th Veh. Technol. Conf. (VTC-Fall)*, Sep. 2021, pp. 1–7.

[68] F. Eckermann, M. Kahlert, and C. Wietfeld, "Performance analysis of C-V2X mode 4 communication introducing an open-source C-V2X simulator," in *Proc. IEEE Veh. Technol. Conf.*, Sep. 2019, pp. 1–5.

[69] (2023). *Spirent GSS9000 GNSS Simulator*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.spirent.com/products/gnss-simulator-gss9000#overview

[70] (2023). *R&S®SMW200A Vector Signal Generator*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.rohde-schwarz.com/us/products/test-and-measurement/vector-signal-generators/rs-smw200a-vector-signal-generator63493-38656.html

[71] (2023). *Spirent GSS6450 - Record & Playback*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.spirent.com/products/gnss-RF-record-and-playback-system-gss6450

[72] (2023). *LabSat 3*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.labsat.co.uk/index.php/en/products/labsat-3

[73] (2023). *Satellite Comms Toolbox*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.mathworks.com/help/satcom/

[74] (2024). *gnssMeasurementGenerator—Simulate GNSS Measurements for Scenarios*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.mathworks.com/help/nav/ref/nav.gnssmeasurementgenerator-system-object.html

[75] (2024). *Simulate GNSS Multipath Effects in Urban Canyon Environment*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.mathworks.com/help/nav/ug/simulate-gnss-multipath-effects-in-Urban-Canyon-Environment.html

[76] (2023). *GPSoft | SatNav Toolbox 3.0*. Accessed: Sep. 6, 2024. [Online]. Available: https://gpsoftnav.com/products/satellite-navigation-satnav-toolbox-3-0/

[77] Y. Li and H. Li, "A collaborative relative localization method for vehicles using vision and LiDAR sensors," in *Proc. 16th Int. Conf. Control, Autom., Robot. Vis. (ICARCV)*, Dec. 2020, pp. 281–286.

[78] G. Zhang, B. Xu, H.-F. Ng, and L.-T. Hsu, "GNSS RUMS: GNSS realistic urban multiagent simulator for collaborative positioning research," *Remote Sens.*, vol. 13, no. 4, p. 544, Feb. 2021. [Online]. Available: https://www.mdpi.com/2072-4292/13/4/544

[79] G. Soatti, M. Nicoli, N. Garcia, B. Denis, R. Raulefs, and H. Wymeersch, "Implicit cooperative positioning in vehicular networks," *IEEE Trans. Intell. Transp. Syst.*, vol. 19, no. 12, pp. 3964–3980, Dec. 2018.

[80] R. Xu, Y. Guo, X. Han, X. Xia, H. Xiang, and J. Ma, "OpenCDA: An open cooperative driving automation framework integrated with co-simulation," in *Proc. IEEE Conf. Intell. Transp. Syst. (ITSC)*, Sep. 2021, pp. 1155–1162.

[81] M. Malinverno, F. Raviglione, C. Casetti, C.-F. Chiasserini, J. Mangues-Bafalluy, and M. Requena-Esteso, "A multi-stack simulation framework for vehicular applications testing," in *Proc. 10th ACM Symp. Design Anal. Intell. Veh. Netw. Appl.*, Nov. 2020, pp. 17–24. [Online]. Available: https://dl.acm.org/doi/10.1145/3416014.3424603

[82] M. Aramrattana, T. Larsson, J. Jansson, and A. Nåbo, "A simulation framework for cooperative intelligent transport systems testing and evaluation," *Transp. Res. F, Traffic Psychol. Behav.*, vol. 61, pp. 268–280, Feb. 2019. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1369847816306635

[83] (2024). *Simulator Technology—VTI.SE*. Accessed: Sep. 6, 2024. [Online]. Available: https://www.vti.se/en/research/vehicle-technology-and-driving-simulation/driving-simulation-visualization-and-simulator-technology/simulator-technology

[84] M. Segata, S. Joerer, B. Bloessl, C. Sommer, F. Dressler, and R. L. Cigno, "Plexe: A platooning extension for veins," in *Proc. IEEE Veh. Netw. Conf. (VNC)*, Dec. 2014, pp. 53–60.

IVO SILVA received the M.Sc. degree in telecommunications and informatics engineering, in 2016, and the Ph.D. degree in telecommunications engineering from the Universities of Minho, Aveiro, and Porto (MAP-Tele Program), in 2022. He is currently a Professor with the Department of Information Systems, University of Minho, and a Researcher with the ALGORITMI Research Centre. He has co-authored several scientific publications, and collaborated in research and development projects in partnership between academia and industry. His research interests include Industry 4.0, indoor positioning and navigation, sensor fusion, vehicle localization, mobile computing, and the Internet of Things.

HÉLDER SILVA received the M.Sc. degree in communications engineering and the Ph.D. degree in electronics and computer engineering from the University of Minho. He is currently an Assistant Researcher with the Centro ALGORITMI, University of Minho. His research interests include GNSS positioning, indoor positioning, wireless sensor networks, and communication protocols.

FABRICIO BOTELHO received the M.Sc. degree in electronics and telecommunications engineering, specializing in automation and control, from the University of Aveiro, Portugal, in 2022. He is currently with Bosch Car Multimedia Portugal, S.A., Portugal, with a focus on software and hardware development. His work includes contributions to advanced driver assistance systems (ADAS), where he is involved in developing technologies aimed at enhancing vehicle safety and automation.

CRISTIANO PENDÃO received the joint Ph.D. degree in telecommunications/computer science engineering from the University of Minho, the University of Aveiro, and the University of Porto, in 2019. He is currently a Professor with the Department of Engineering, University of Trás-os-Montes and Alto Douro, Portugal. He is also a Full Research Member of the ALGORITMI Research Centre, University of Minho. He has actively participated in innovation and commercial projects within the industry. He has co-authored several scientific publications. His research interests include mobile computing, positioning and navigation, computer vision/perception, and machine learning.