A SUMO-Based Hardware-in-the-Loop V2X Simulation Framework for Testing and Rapid Prototyping of Cooperative Vehicular Applications

Zsolt Szendrei(✉), Norbert Varga, and László Bokor

MediaNets Laboratory, Commsignia - BME HIT V2X Communication Research Group, Department of Networked Systems and Services, Budapest University of Technology and Economics, Műegyetem rakpart 3-9, Budapest 1111, Hungary
{szendrei, vnorbert, bokorl}@hit.bme.hu

Abstract. Vehicle-to-Anything (V2X) technologies aim at providing globally standardized communication tools to efficiently transmit information between all players of the transportation ecosystem. Based on such extensive data exchange between traffic objects V2X helps to create an advanced domain of transportation called Cooperative Intelligent Transportation Systems (C-ITS) where an ever-growing scale of cooperative vehicular applications/services facilitate enhanced safety and comfort on the road and lead towards fully automated transportation in the future. C-ITS relies on a complex architecture consisting of a cross-layer optimized sophisticated protocol stack, hybrid radio access solutions, computing algorithms, decision schemes, special interfaces, internet-of-things (IoT) integration, etc. Moreover, C-ITS and the support of cooperative use-cases demand high reliability, enhanced Quality of Service/Quality of Experience (QoS/QoE), and rock-solid hardware/software implementations working efficiently and securely even in the most complicated environments including extreme traffic circumstances or unpredictable actions. Therefore, deliberate system testing is a crucial and strategic process, especially when we are closing to the wide-scale deployment phase of real-life C-ITS solutions. Our proposed hardware-in-the-loop (HiL) V2X simulation framework was designed to offer a cost-efficient and simple toolset for testing and rapid prototyping of cooperative vehicular solutions by partially replacing costly, time-consuming and oftentimes dangerous field tests with an easy to install, tabletop laboratory test- and development suite.

Keywords: Vehicle-to-Anything (V2X) communication
Hardware-in-the-Loop (HiL) simulation · Rapid prototyping and testing
Traffic simulation · Cooperative applications/services
Cooperative Intelligent Transportation Systems (C-ITS)

K. Jármai and B. Bolló (Eds.): VAE 2018, LNME, pp. 426–440, 2018.
https://doi.org/10.1007/978-3-319-75677-6_37

1 Introduction

Intelligent Transport Systems (ITS) support transportation of goods and humans with the help of various Information and Communication Technologies (ICT) in order to improve the performance of traffic management and warning systems, the safety, productivity and efficiency of transportation services [1]. The ITS technology evolution can get more momentum if the vehicles/infrastructure communicate with each other and exchange relevant information using Cooperative ITS (C-ITS) or Vehicle-to-Anything (V2X) solutions [2, 3]. The emergence of direct Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communications and the proliferation of cooperative vehicular applications like [4–6] increase the level of awareness regarding vehicles' surroundings and therefore V2X serves as one of the main technological enablers for autonomous transportation [7]. As evolving V2X solutions tend to be designed to meet self-driving car and autonomous driving requirements it becomes more and more important to efficiently support test and development activities of V2X technologies, applications and services. In order to foster rapid prototyping and proficient testing of V2X software in a microscopic traffic simulation environment we designed and implemented a V2X hardware-in-the-loop (HiL) simulation framework around the well-known free and open traffic simulation suite called SUMO - Simulation of Urban Mobility [8]. Our open and freely available framework allows the user to run highly realistic simulations of cooperative and autonomous vehicle functions by assigning real-life V2X hardware/software stacks to simulated road vehicles, pedestrians, smart traffic lights, etc. in an OpenStreetMap [9] initiated SUMO scenario, and by controlling the interactions with the TraCI interface [10] through a special orchestrator module. The paper introduces the details of the overall HiL simulation framework, and by detailing V2I/I2V and V2V cooperative vehicular application simulation scenarios highlights the main benefits and also the limitations of the proposed scheme.

The remainder of the paper is structured as follows: Sect. 2 summarizes the state-of-the-art research in the area of V2X HiL simulation, with particular focus on the existing simulation tools and environments. Section 3 overviews the details of our proposed simulation framework by introducing all the building blocks and their interoperation, together with example scenarios, which are presented in Sect. 4. The paper is concluded in Sect. 5, pointing out the potential continuation of our work-in-progress efforts within this research topic.

2 Related Work

Despite the fact that field operational tests and large-scale deployments are already going on, V2X development and performance evaluation is still in a strong need for simulation experiments. Simulation of V2X systems, architectures, protocols, applications or services has a very broad literature; there are several dozen available simulation tools, frameworks and solutions to evaluate vehicular communication. Authors of [11, 12] provide two comprehensive, broad surveys on the prominent simulation platforms and their components like graphical user interface, popularity, ease of use, input requirements, accuracy, etc. However, the list of V2X simulators providing hardware-in-the-loop capabilities is much shorter. And when it comes to non-commercial, open or free V2X simulators with HiL support, the list becomes empty, at least according to the best of our knowledge. In the below sections we shortly introduce the related work by surveying the identified commercial products and related publications in the domain.

In [13] authors introduce a connected and autonomous vehicle hardware-in-the-loop simulator designed primarily for developing automated driving algorithms by performing HiL simulations of connected and autonomous driving functions. The HiL simulator of their proposed framework consists of several commercial elements: a dSPACE Scalexio system [14] which runs CarSim Real Time [15] with moving objects and ADAS sensors, and is connected to a dSPACE Microautobox electronic control unit [16] and two V2X radio adapters for V2X communication. The dSPACE Scalexio is a specialized, scalable, high-performance real-time computing hardware for demanding applications and comprehensive, precise, and fast I/O capabilities. Carsim Real Time provides accurate, detailed, and efficient methods for simulating the performance of passenger vehicles and light-duty trucks, and facilitates physical testing and simulation joined in real-time to evaluate hardware being driven by a software vehicle dynamics model. The dSPACE Microautobox is a real-time system for performing fast function prototyping and able to operate without user intervention, just like any electronic control unit (ECU). Authors included this control unit in an actual autonomous vehicle (a Ford Fusion Hybrid SE) that has been completely automated in the research work. V2X hardware adapters emulate IEEE 802.11p-based communication between the vehicle and all the other objects in the simulation. Authors focus on the ECU-based implementation of Lane Keeping and Cooperative Adaptive Cruise Control (CACC) applications, V2X has only marginal roles in [13]. However, their applied dSPACE V2X Blockset for Simulink [17] is ready to allow developers to connect their applications directly to the appropriate communication blocks, where implemented protocols and software layers perform further steps, such as encoding and decoding V2X messages, and enable the communication to a real-life V2X hardware adapter. As a standalone feature is does not allow to apply synthetic traffic into the V2X simulation, but with other dSPACE elements and CarSim it is achievable.

Another CarSim based approach is [18] where authors presented a pedestrian collision warning and avoidance system for road vehicles based on V2X communication. A HiL simulation setup was created to validate the proposed automated driving vehicle model in Carsim Real Time [15] together with two V2X modems emulating the vehicle and pedestrian communications during the development and evaluation.

ADAS iiT, a collaboration of four National Instruments (NI) Alliance partners also offer a commercial HiL simulator for V2X [19] based on the products of S.E.A. Datentechnik GmbH [20] and NI technology and platform [21]. Their V2X test system solutions are designed as a modular scheme of building blocks with clearly defined interfaces based on NI hardware and software, S.E.A. V2X communication interfaces and GNSS simulation products. The solution is flexible: tests of single devices or complete systems in functional or HiL-testing applications are also achievable, especially in the V2X sensor fusion domain. Intra-vehicle communication, position information management and scenario generation are both available.

IPG Automotive GmbH recently advertised [22] that their HiL supporting CarMaker simulation software [23] is applied for testing systems in Germany's Providentia V2X project [24]. Germany's A9 highway will be realistically transferred to CarMaker's virtual world, then real-traffic situations and objects are virtually modeled in real time. If the interaction of sensors, V2X components, and advanced driver assistance systems (ADAS) will also work in the simulations as planned, the real-time data can then be displayed on in-car systems while the connected vehicles are being tested in actual traffic scenarios. Not much details are available on this effort, but Providentia continues until mid-2019 [25], therefore further information is to be published in the near future.

Spirent Communications Inc. developed ITS G5 and WAVE DSRC conformance testing tools [26, 27] based on real V2X hardware and TTCN-3 support. Moreover, in partnership with Tata Elxsi they created a V2X Emulator [28], which is also a commercial product to be used for validation and performance benchmarking of V2X applications running on V2X ECUs. By applying GNSS and radio channel emulators this tool also has the ability to bring real-world traffic scenarios into the laboratory. However, there is no available information on the supported traffic simulation complexity.

Authors of [29] present the HI^{2}LS framework consisting of physical WAVE communication devices, smart phones with a road safety application, human-interactive devices of a steering wheel and a brake pedal, and the simulation models of car and road traffic environments based on PreScan [30] which is a commercial simulator allowing virtual vehicles emulating real ones driven by human drivers, and standard C-ITS message set based communication. Thanks to PreScan and the integrated human-interactive devices, HI^{2}LS provides a very broad scale of features but similarly as all the above solutions it is also based on a commercial product and expensive hardware-software elements making its wide-scale application doubtful for certain use-cases.

3 The Proposed HW-in-the-Loop V2X Simulation Framework

3.1 General Architecture

The proposed simulation system is basically a coordinated cooperation of different, freely available, open-source components (except the OBU/RSU V2X devices). The overall framework is available on the project's GitHub website [31]. Figure 1 summarizes the general architecture and the following sub-chapters detail all the integrated modules and their interfaces.

The core entity is the Orchestrator, which manages all other elements. There is one SUMO [8] simulator instance for visualization and realistic traffic generation/simulation. The amount of gpsfake instances (implementing the location information provisioning subsystem) depends on the quantity of the attached real HW-based OBU/RSU (On Board Unit/Road Side Unit) devices. Each OBU/RSU has a specific gpsfake [32] instance modified with appropriate extensions and belonging to the OBU's own GPS (Global Positioning System) engine, and continuously produces essential location data required by the real-life HW/SW V2X protocol stack of the OBU/RSU units. These devices also run V2X applications, which simulator users would presumably like to develop and test together with the HW/SW V2X stack itself using the realistic simulated traffic environment. The OBU is assigned to selected vehicles of SUMO, collects the vehicle's simulated

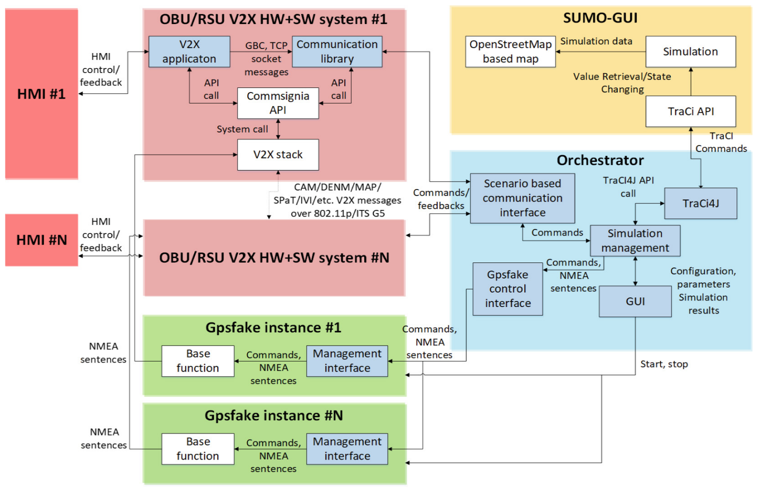

Fig. 1. The architecture of our designed HiL V2X simulation framework.

sensor data, could warn the driver using the Human-Machine Interface (HMI), cooperate with autonomous driving functions and of course communicates with other V2X devices of a particular simulation scenario. RSUs are assigned to smart traffic lights, traffic junctions, provide highway or smart city coverages, etc. helping to introduce the infrastructure's point of view for the C-ITS ecosystem, and such communicating with the reachable V2X devices. The amount of attached OBU/RSU devices into our framework is not restricted, however, it depends on the actual scenario and the resources of the computer which runs the Orchestrator (10–20 V2X hardware systems with moderate packet sending rate could be easily integrated into our simulation loop on an average PC). Relying on the used V2X protocol stack and the included API any required HMI can be implemented e.g. based on an Android tablet.

In our framework the communication between the Orchestrator and the supplemental modules is based on TCP sockets. The TraCI [10] and the gpsfake interfaces require commands and parameters in sentences split by separators, while the communication with the V2X devices is performed through JSON type messages. The simulation framework does not provide a general interface for the HMI: OBU-driver interactions or RSU control and feedback can be implemented with the available API of the applied V2X device.

Macroscopic Traffic Simulator Subsystem: SUMO-GUI
SUMO (Simulation of Urban MObility) is a free and open microscopic traffic simulation tool [8]. The microscopic nature of SUMO means that all the vehicles are controlled separately, all of them have specific source and destination, physical properties, etc. The simulation is performed in discrete time and the speed of the vehicles depends on the rules and the traffic model in which the vehicles flow. We integrated SUMO into our framework by using the TraCI interface providing flexible control over this realistic traffic simulation and visualization tool.

The TraCI (Traffic Control Interface) [10] is an interface, which makes it possible to query SUMO simulation data about the vehicles, route, traffic lights etc. Moreover, it also allows to modify the simulation parameters regarding to the given set of commands. Therefore, outside of the SUMO simulation (i.e. from the overall HW-in-the-loop simulation framework point of view) we can modify the route of any individual vehicles, their speed, etc. It is a very useful feature used by our Orchestrator module to interact with the traffic simulation. For example, if the traffic density becomes high an OBU can request a new route and we can calculate it for that particular vehicle, the related communication can be performed between a responding RSU and this particular OBU, and the resulted new heading, speed, etc. can be forced according to the actual traffic circumstances.

In our framework SUMO is an essential part as it also produces location data for all the involved C-ITS stations. Moreover, the GUI of the SUMO is used to visually follow the map, the movement of vehicles, and in general helps to visualize the simulation. Our framework relies on the SUMO's OpenStreetMap engine and uses the imported OpenStreetMap data. The Orchestrator's User Interface can be used to configure the simulation and assign gpsfake instances to the simulated vehicles.

Real-Life Compatible Location Data Provision Subsystem: gpsfake
One of the main goals of our HW-in-the-loop V2X simulation framework is to integrate real vehicular communication devices, so compatible location/navigation data provision is a crucial requirement. Almost all V2X devices on the market have the feature of supporting gpsd connection as an alternative location data source to the real GPS hardware. The gpsd software is a monitor daemon [33] collecting GPS data from real GPS devices or synthetic GPS feeds. Besides collecting the data, the daemon also makes available to other programs in order to share the information through e.g. a socket interface. These programs can subscribe to the daemon, which immediately sends the collected location data over TCP, so the applications can believe that they are connected to a real GPS device and received real-life GPS feed.

The gpsfake [32] is a test harness between the gpsd and the client applications. It launches a gpsd instance that believes it communicates with a real GPS device. The navigation data is read from a pre-recorded log file in any supported format: NMEA, SiRF, TSIP or Zodiac. The disadvantage of gpsfake from our point of view is that it works statically (it processes and reads only file-based, predefined GPS trace inputs), meaning that it is not possible to feed the gpsd with dynamic navigation data (i.e. modify the path of a vehicle). Therefore, it was necessary to overcome this obstruction and make it possible to dynamically change the location data when it is necessary e.g. thanks to V2X communication based traffic information exchange. In order to achieve this goal, we added a novel management interface extending the original gpsfake functionalities and making the missing dynamic location information provision and navigation possible. This newly introduced management interface awaits commands, NMEA sentences, and therefore it can feed gpsd with dynamically arriving location data from the SUMO-GUI subsystem of our framework. The implemented novel gpsfake commands:

- *stop/go*: The command pair makes enable to stop the vehicle for a while (zero speed, constant position) in case of need.

- *manual*: After executing this command the gpsfake instance will accept NMEA sentences trough the management interface and then feeds them to the vehicle using gpsd.
- *file*: A gpsfake instance accepts a log file, which contains sentences according to an acceptable GPS format. With this command we can dynamically upload a new log file to gpsfake, which file will be then used as a novel source of GPS information.

Central Simulation Management and Control Subsystem: Orchestrator
The Orchestrator is the main part of our simulation framework as it makes possible to all the different components to communicate with each other, share the information, configure the simulation parameters and schedule the simulation. The Orchestrator is written in Java 8 with JavaFX [34] graphical user interface. As gpsfake only supports Unix based operating systems (Linux, and the BSD family), it is also the requirement for the Orchestrator to run. The UI handles the simulation parameter set up, the SUMO and gpsfake configuration, the scenario selection/configuration and the result collection/visualization tasks.

A scenario describes the available commands, which the V2X application can send and the orchestrator can accept. The communication between the vehicular communication devices (OBUs/RSUs) and the orchestrator is handled with JSON type messages and the required logic must be implemented inside the V2X application. These control messages contain the command, the parameters and the assigned vehicle's ID.

SUMO is a discrete simulation tool, but the OBU/RSU devices communicate to each other and send commands to Orchestrator in real time, so the Orchestrator must translate between these two worlds. We follow a simple but efficient approach: control messages are inserted into a queue, and in every discrete simulation steps these arrived commands are processed. Most of the commands are TraCI commands, so it is necessary for the Orchestrator to have an appropriate connection to the SUMO. For this purpose the TraCI4J [35] library is used. This library is in alpha version, there are a lot of missing unavailable TraCI commands, so we completed several ones. Fortunately, TraCI4J is also open, so it was possible to complete the free library with the required commands (e.g. change global effort information, edge effort information, last step mean speed, etc.).

A crucial part of the Orchestrator is the real-time location management and navigation for the HW-in-the-loop simulated V2X OBU/RSU devices. For this purpose we applied a special data structure in our implementation to support fast lookup and search functions in the map database. Unfortunately, there are some static map data, which cannot be queried form SUMO through TraCI, for example road segment's (edge) position, length and width, i.e. those parameters, which describe the geometry of an edge. Edges are straight road segments, basically the main building blocks of any SUMO map, which make easy to use map elements. Edge descriptor information required for the planned on-the-fly search functions can be used to generate rectangles, and these rectangles will be put into the special data structure called R-Tree [36], which efficiently supports advanced spatial search functions (e.g. search by coordinates). The main advantage of this data structure is that it makes possible to find a given route segment so fast, that we can rely on it even in real-time circumstances. Of course, the creation of the R-Tree from the SUMO data is a slow process: all road segments are converted into rectangles regarding to their position, length, and width information. Fortunately, this resource consuming function must be executed only once, at the simulation startup process. Then, during runtime, appropriate R-Tree algorithms can be used on the data structure efficiently. The R-Tree adds more rectangles to overlap other rectangles, and it makes possible to build the tree itself (Fig. 2 upper part). When we are looking for a particular position (coordinates) in this tree, the algorithm checks whether the given point is contained by the rectangle (Fig. 2 lower part), and regarding this, the algorithm will eventually reach the leaf element, just be checking the containment continuously. The wanted road segment is located in the leaf element.

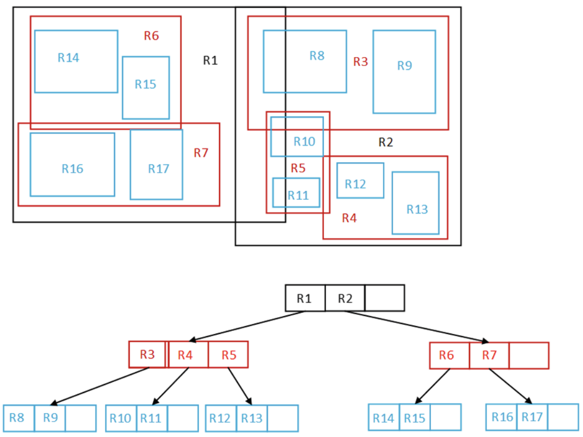

Fig. 2. Containment and overlapping relationships in R-Tree (upper part) and the created R-Tree structure from rectangles (lower part).

The area of the used map (Fig. 3) is 5.69 km^2, and after we stored it in an R-Tree data structure it contains 10,922 rectangles (Fig. 4). This amount of rectangle can cover all edges on the map. To store the edges in R-Tree, we use the JSI (Java Spatial Index) RTree Library [37]. The average time to find an edge regarding to a position in the constructed tree is 1.36 ms, while the average search time in a list is 4.37 ms. In both cases the search algorithm examines point-rectangle containment. In addition, the data structure is filled with the computed rectangles before the simulation starts. In worst case all the n element is examined, while the R-Tree complexity is O(log n). Moreover, the R-Tree algorithm supports the nearest neighbor search, so if there is no match, it can give back the nearest rectangle as a result.

V2X Device Adaptation: OBU/RSU V2X Application
This part of the framework deals with the adaptation of the real device into the HW-in-the-loop system. We use Commsignia OB2 devices [38] for OBU/RSU communication application testing and development, but any other V2X HW/SW could be integrated. One of the purposes of this simulation tool is to make it easier to use the V2X devices in laboratory environments, where there are no real GPS feed, no movement, no real vehicles and traffic circumstances available.

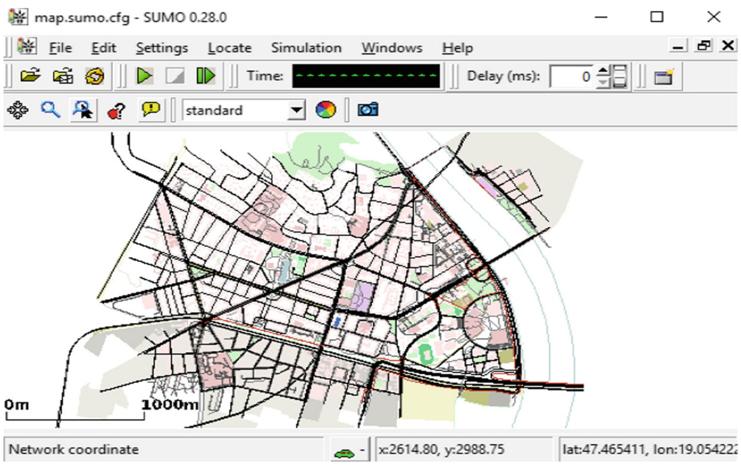
map.sumo.cfg - SUMO 0.28.0
File Edit Settings Locate Simulation Windows Help
Time:
Delay (ms): 0
standard
0m
1000m
Network coordinate
x:2614.80, y:2988.75
lat:47.465411, lon:19.05422

Fig. 3. The used SUMO map, imported from Open Street Map.

With SUMO support the gpsfake/gpsd can feed the V2X devices with the so synthetic GPS information, so we are able to develop and test any application in realistic traffic scenarios. The only requirement of the adaptation is to implement an interface in the V2X application of the OBU/RSU device, that keeps the connection with the Orchestrator using our well-defined interface and commands. Then the OBU/RSU devices communicate with each other using the standardized message exchange sequences (e.g. CAM, DENM, MAP, SPAT, etc.), and if it required by the simulation they can communicate with the navigation system (i.e. the Orchestrator).

V2X Device Outputs: Human-Machine Interface
The Human-Machine (HMI) interface is a component, which aims to provide useful information for the driver and thus it provides real-life outputs of the V2X devices through e.g. an Android tablet based system. The V2X application processes all the incoming messages e.g. CAM, DENM, etc. and builds its actual level of cooperative awareness, creates a view of the neighboring vehicles, infrastructures elements. There are a lot of traffic safety applications, which can use these messages. These applications can warn the driver about a dangerous situation and if it is possible the V2X device can initiate warning messages, later even intervention. These notifications can be useful, they might prevent a collision, accidents, etc. Some useful examples for applications with HMI can be found in [39], but of course any HMI solution can be applied here. RSU monitoring/tracing/management can also be imagined as a HMI module.

3.2 Simulation Workflow

To create a new scenario and test a new V2X solution in our simulation framework the first step of all is that we should implement the appropriate V2X application. That application can be an average V2X app with standardized interfaces, so it is important to work in the similar way as in a real C-ITS environment. However, our interface – connecting the app to the Orchestrator (V2X-Orchestrator or VO interface) – is compulsory, it must be added.

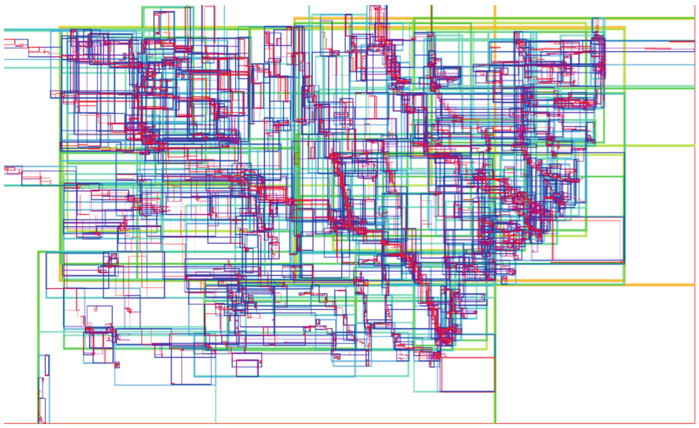

Fig. 4. The generated R-Tree from the used SUMO map.

Before implementing a particular scenario, the required commands must be designed according to the VO interface specification. Regarding to the V2X message exchange and results the Orchestrator will translate them into Orchestrator commands. When a new VO command arrives to the Orchestrator, it checks whether it is a valid command or not. If yes, the system generates a proper Orchestrator command according to the VO command. So, after the VO commands are defined, the next step is to implement the Orchestrator commands. A command can make a query or a change to a SUMO object state through the TraCI. It can reach the R-Tree to get static information about the map, moreover an Orchestrator command can send messages and/or VO commands to other simulation elements, V2X devices. After the command instance is created, it will be put in a queue and will be waiting for the next simulation step to be executed. Every managed entity has a separate queue and in every simulation step these objects process their tasks. Managed devices are the observed, supervised objects (usually the V2X devices), which can be vehicles, traffic lights, whatsoever but there must be a SUMO representation of them and their V2X application must also support the particular function.

By defining your own scenarios with the above steps you can use the simulation tool to get feedbacks during the V2X application development. Test cases can be created based on the scenario to verify and/or demonstrate appropriate operation in different traffic and environmental circumstances before the application would be tested in a real world environment, without any risks.

The weakness of our proposed tool and its workflow is that the scheme strongly relies on the SUMO TraCI interface: any scenario planning requires to check whether the expected function could be realized using TraCI or not. Moreover, both the V2X application and the Orchestrator require to introduce the proper interfaces and functionalities, which creates development overhead.

4 Example Simulation Scenarios

4.1 TMC Initiated Rerouting (V2I/I2V Communication)

In order to demonstrate the operation of our proposed simulation framework, we implemented two example scenarios. The first scenario shows how V2X communication supports dynamic rerouting of a particular vehicle in a traffic congested area. The simulation tool handles the task of the congestion detection using a separate module inside the Orchestrator. The scenario consists of four main elements: the Navigation system, OBU, RSU and TMC (Traffic Management Centre) (Fig. 5).

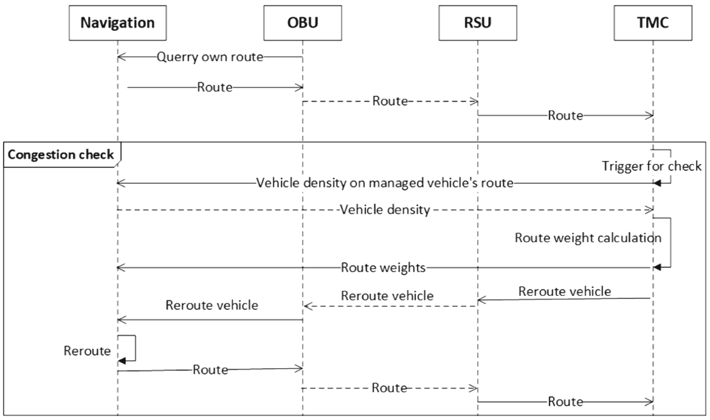

Fig. 5. Sequence of TMC initiated rerouting.

The Navigation system is part of vehicle but implemented also within the Orchestrator; this component can communicate with the SUMO, it can change the vehicles route, query the state, etc. The OBU is assigned to a SUMO simulated vehicle, which is set to go from A to B. When it needs map data or it wants to change something in the current route, it will communicate with the Navigation system. The RSU is an entity, which implements the infrastructure side, emulates the system of RSUs controlling a larger area of the city. The RSU communicates with the vehicle under test using standard C-ITS messages (e.g. TPEG 2). In case of need the Navigation system component initiates queries to the TMC or receives periodic traffic information. The TMC exchanges traffic information gathered from SUMO through the Orchestrator, and periodically checks traffic conditions for registered vehicles (i.e. vehicles with OBUs under test or evaluation), and performs reroute in case of need. The traffic is generated by SUMO (specified in the SUMO configuration file describing the routes of vehicles). In this scenario the predefined routes are mainly concentrated on that road, where the vehicle under evaluation plans to go. In that way the TMC's ability to advise rerouting information to the vehicle can be tested as well.

By relying on real-time traffic information TMC can decrease the travel time. In our framework the TMC is part of the Orchestrator, but logically it is a separated element maintained by the city road infrastructure operator. The TMC does not continuously receive traffic condition information, it should query that from the Orchestrator, which can give detailed status report regarding given roads or road sets. SUMO supports only one connection through the TraCI, so the TMC gathers the traffic information from the Orchestrator module managing that connection. If the TMC finds out that the traffic condition is getting worse harming the fast movement of a considered route or monitored vehicles, than it can notify the considered vehicles, which then ask the Navigation system for a reroute. If the Navigation system finds a better route, than it sends the recalculated path to the considered vehicles and also informs the TMC about the changed paths.

4.2 OBU Initiated Rerouting (V2V Communication)

The second scenario is simpler as it leaves out the infrastructure part from the sequence and uses only V2V communication. The setup consists of three main components: the Navigation system and two OBUs.

At the beginning of this scenario two vehicles with assigned OBUs start their route (Fig. 6). Few seconds after the simulation started, one of the two monitored vehicles (with assigned OBUs) stops and it begins to send Stationary Vehicle Notification DENM messages. When the other OBU receives the notification, its V2X application constructs a message with the stationary vehicle's position and sends a query to the Navigation module to ask for a reroute if it is necessary. The Navigation module checks whether the given position is in the vehicle's route or not and notifies the vehicle about the result and performs rerouting if required.

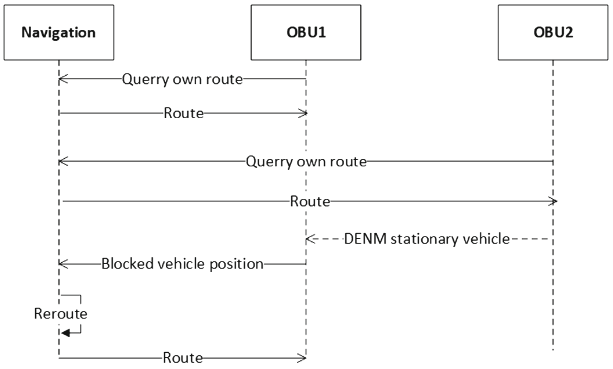

Fig. 6. Sequence of OBU initiated rerouting

This scenario demonstrates both the V2V communication effectiveness and a simple use-case of our simulation framework. Without the V2V exchange the first vehicle would stuck in a traffic jam caused by the second one, and its travel time would increase significantly. With V2V cooperative awareness, the vehicle incorporating OBU1 can quickly reroute, resulting in a better detour towards its destination. Application prototyping and testing is easily achievable.

5 Conclusions

In this paper we proposed a HW-in-the-loop V2X simulation framework, which supports real V2X OBU/RSU devices with real C-ITS protocol stacks and APIs, and makes possible to develop and test real V2X applications in realistic traffic conditions provided by a laboratory simulation/evaluation environment based on SUMO. As a part of our future work we would like to improve the framework with more generic interfaces, which support more commands and more sophisticated interaction between the V2X devices and the Orchestrator. We would like to develop a statistical and result analysis module, which could help us to easily see the differences between V2X solutions, so we can analyze how effective could be the application of a particular V2X technology. Furthermore, we would like to add other extensions to make the evaluation setup and assessment as general and efficient as possible, because it gives the chance to widen the range of applicable V2X devices. Moreover, with the help of the proposed framework we plan to analyze the characteristics and performance of different C-ITS applications/services in real traffic circumstances focusing on smart city environments especially where the proliferation of V2X enabled vehicles is low (e.g. use-cases with environmental perception messaging).

Acknowledgements. The work leading to these results has been supported by Commsignia Ltd. and by the Sándor Csibi Scholarship of the Pro Progressio Foundation. The authors are also grateful for the cooperation, assistance and support of the Újbuda SMART 11 Nonprofit Kft. and Magyar Telekom.

References

1. Grant-Muller, S., Usher, M.: Intelligent transport systems: the propensity for environmental and economic benefits. Technol. Forecast. Soc. Change **82**, 149–166 (2014)
2. Sjoberg, K., Andres, P., Buburuzan, T., Brakemeier, A.: Cooperative intelligent transport systems in Europe: current deployment status and outlook. IEEE Veh. Technol. Mag. **12**(2), 89–97 (2017)
3. Festag, A.: Cooperative intelligent transport systems standards in Europe. IEEE Commun. Mag. **52**(12), 166–172 (2014)
4. Chen, D., Ahn, S., Chitturi, M., Noyce, D.: Truck platooning on uphill grades under cooperative adaptive cruise control (CACC). Transp. Res. Procedia **23**, 1059–1078 (2017)
5. Wan, N., Vahidi, A., Luckow, A.: Optimal speed advisory for connected vehicles in arterial roads and the impact on mixed traffic. Transp. Res. Part C Emerg. Technol. **69**, 548–563 (2016)
6. Jiménez, F., Naranjo, J.E., Anaya, J.J., García, F., Ponz, A., Armingol, J.M.: Advanced driver assistance system for road environments to improve safety and efficiency. Transp. Res. Procedia **14**, 2245–2254 (2016)
7. Krasniqi, X., Hajrizi, E.: Use of IoT technology to drive the automotive industry from connected to full autonomous vehicles. IFAC-Pap. **49**(29), 269–274 (2016)
8. Behrisch, M., Bieker, L., Erdmann, J., Krajzewicz, D.: SUMO - simulation of urban mobility: an overview. In: The Third International Conference on Advances in System Simulation, SIMUL 2011, pp. 63–68 (2011)

9. Ramm, F., Topf, J., Chilton, S.: OpenStreetMap: using and enhancing the free map of the world. UIT Cambridge, Cambridge (2011)
10. SUMO TraCI (Traffic Control Interface). http://sumo.dlr.de/wiki/TraCI. Accessed 19 Jan 2018
11. Sommer, C., Härri, J., Hrizi, F., Schünemann, B., Dressler, F.: Simulation tools and techniques for vehicular communications and applications. In: Campolo, C., Molinaro, A., Scopigno, R. (eds.) Vehicular ad hoc Networks, pp. 365–392. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-15497-8_13
12. Martinez, F.J., Toh, C.-K., Cano, J.-C., Calafate, C.M.T., Manzoni, P.: A survey and comparative study of simulators for vehicular ad hoc networks (VANETs). Wirel. Commun. Mob. Comput. **11**, 813–828 (2011)
13. Gelbal, Ş.Y., et al.: A connected and autonomous vehicle hardware-in-the-loop simulator for developing automated driving algorithms. In: 2017 IEEE International Conference on Systems, Man and Cybernetics (SMC), pp. 3397–3402 (2017)
14. dSPACE SCALEXIO. https://www.dspace.com/en/ltd/home/products/hw/simulator_hardware/scalexio.cfm. Accessed 19 Oct 2018
15. CarSim Mechanical Simulation. https://www.carsim.com/products/carsim/. Accessed 19 Oct 2018
16. dSPACE MicroAutoBox. https://www.dspace.com/en/pub/home/products/hw/micautob.cfm. Accessed 19 Oct 2018
17. dSPACE V2X Blockset for Simulink. https://www.dspace.com/en/inc/home/products/sw/impsw/v2xsolution.cfm. Accessed 19 Oct 2018
18. Gelbal, S.Y., Arslan, S., Wang, H., Aksun-Guvenc, B., Guvenc, L.: Elastic band based pedestrian collision avoidance using V2X communication. In: 2017 IEEE Intelligent Vehicles Symposium (IV), pp. 270–276 (2017)
19. ADAS iiT - V2X/GNSS open and closed loop (HIL) test for V&V. https://www.adas-iit.com/v2x-gnss-simulation/v2x-test/v2x/. Accessed 19 Oct 2018
20. S.E.A. Datentechnik GmbH - Customized HIL-Test systems for V2X and ADAS Verification. https://www.sea-gmbh.com/v2x/v2x/. Accessed 19 Oct 2018
21. V2X/802.11p Validation and System Test Solution on NI platform. http://partners.ni.com/solutions/solution/39/v2x-802-11p-validation-and-system-test-solution. Accessed 19 Oct 2018
22. IPG's simulation software testing systems for Germany's Providentia V2X project. http://www.traffictechnologytoday.com/news.php?NewsID=88744. Accessed 19 Oct 2018
23. CarMaker: virtual testing of automobiles and light-duty vehicles. https://ipg-automotive.com/products-services/simulation-software/carmaker/. Accessed 19 Jan 2018
24. BMVI - Providentia Projekte. http://www.bmvi.de/SharedDocs/DE/Pressemitteilungen/2016/205-dobrindt-foerderbescheide-digitale-testfelder.html. Accessed 19 Jan 2018
25. Providentia - Proaktive Videobasierte Nutzung von Telekommunikationstechnologien in Innovativen Autobahn-Szenarien. https://www.fortiss.org/forschung/projekte/providentia/. Accessed 19 Jan 2018
26. TTsuite-WAVE-DSRC – Spirent. https://www.spirent.com/Products/TTworkbench/TTsuites/WAVE-DSRC. Accessed 19 Oct 2018
27. TTsuite-ITS-G5 – Spirent. https://www.spirent.com/Products/TTworkbench/TTsuites. Accessed 19 Jan 2018
28. Spirent V2X Emulator. https://www.spirent.com/Products/V2X-Emulator. Accessed 19 Jan 2018
29. Song, S., et al.: Demo: human-interactive hardware-in-the-loop simulator for cooperative intelligent transportation systems and services. In: 2015 IEEE Vehicular Networking Conference (VNC), pp. 169–170 (2015)

30. PreScan - Simulation of ADAS & active safety. https://tass.plm.automation.siemens.com/prescan. Accessed 19 Jan 2018
31. SUMO-based Hardware-in-the-Loop V2X Simulation Framework Source Codes. https://github.com/szzso/HIL-V2X-Simulation-with-SUMO-Support, Accessed 18 Jan 2018
32. Raymond, E.S.: gpsfake—test harness for gpsd, simulating a GPS. http://catb.org/gpsd/gpsfake.html. Accessed 19 Jan 2018
33. gpsd—a GPS service daemon. http://catb.org/gpsd. Accessed 19 Jan 2018
34. Docs.oracle.com: JavaFX Overview (Release 8)
35. TraCI4 J. https://github.com/egueli/TraCI4J. Accessed 19 Oct 2018
36. Guttman, A.: R-trees: a Dynamic Index Structure for Spatial Searching. SIGMOD Rec. **14** (2), 47–57 (1984)
37. JSI (Java Spatial Index) RTree Library. https://github.com/aled/jsi. Accessed 19 Jan 2018
38. Commsignia Ltd. http://www.commsignia.com/. Accessed 19 Oct 2018
39. Popescu-Zeletin, R., Radusch, I., Rigani, M.A.: Vehicular-2-X Communication: State-of-the-Art and Research in Mobile Vehicular Ad Hoc Networks, 1st edn. Springer, Heidelberg (2010)