3D Driving Simulator with VANET Capabilities to Assess Cooperative Systems: 3DSimVanet

Florian Michaeler and Cristina Olaverri-Monreal* *Member, IEEE*

***Abstract*—Vehicular systems that intend to aid the driver to increase road safety and reduce road accidents might differ in the levels of safety provided if they are based on different functioning and visualization concepts. In this paper we present a flexible and adjustable 3-dimensional driving simulator based on Open-StreetMap (OSM) data that integrates Vehicular Ad Hoc Network (VANET) communication capabilities to assess different information paradigms in a lab-controlled environment. Results regarding the proof of concept of the simulator to test new in-vehicle technology validated the proposed approach. The experiments conducted show discrepancies in the driver's environment perception that were determined by the system used and how the information was provided to the driver.**

I. INTRODUCTION

The main aim of Advanced Driving Assistance Systems (ADAS) and other similar systems that aid the driver is to increase road safety and reduce road accidents. But do systems with different functioning concepts provide the same level of safety? What is the best method to compare such systems? In this paper we present a flexible and adjustable driving simulator platform to address the questions that arise from the system presented in [1] as a proof of concept. Whether on the rear of the leading vehicle or directly transmitted via Vehicular Ad Hoc Network (VANET) into the following vehicle, does information display modality regarding the recommended safety distance affect driver response in a simulated environment?

The constant progress in computer science and telecommunication technologies has contributed to rapid development in the area of Vehicle to Vehicle (V2V) communication and more and more ADAS are being implemented relying on this communication technology. As new in-vehicle assistants need to be evaluated in a lab-controlled environment before performing a field test, we present in this work a simulator which is capable of handling both the VANET communication and vehicle movement at once. We additionally evaluate the effects of V2V communication through the impact on drivers of VANET-based ADAS against DAS in unequipped vehicles.

The next section considers related work in simulation approaches, including V2V communication. Section III describes the development of the simulation platform. Section IV presents the implementation of the simulation environment. Section V describes routing and behavior processes of the simulation-controlled vehicles. Section VI presents a detailed description of the implementation of the ad hoc network communication. Section VII reports on the validation results of the development software. Finally, Section VIII concludes the paper.

*Corresponding author

University of Applied Sciences Technikum Wien, Department of Information Engineering and Security, Vienna, Austria; florian@michaeler.at, olaverri@technikum-wien.at

II. RELATED WORK

In contrast to top down view driving simulators three-dimensional (3D) simulators replicate the width, height and depth of our physical environment. Some 3D driving simulators, aim at increasing road situational awareness to reduce distraction and increase safety through educational or serious games in specific challenging situations [2]. To achieve a realistic effect, simulation tools need to take into account all interactions between the driver, vehicle and traffic as in the approach in [3] where these elements were modeled and delineated by interfaces in a modular program structure. Approaches related to 3D simulation have also been presented by the Deutsches Zentrum für Luft und Raumfahrt (DLR) in the context of the development of the Simulation of Urban MObility (SUMO) software and related plugins [4]. Special attention was given to the development of 3D landscapes and simulations based on available map and height information [5]. The authors discussed some of the main problems that are caused by using a procedural generated environment instead of a hand-crafted environment.

VANET simulations are commonly performed to evaluate different communication protocols and their performance [6]. Most of the related literature builds on the main discrete network event simulators NS-3 [7] or Omnet++ [8]. Both simulators are customizable and widely used in research. Other simulators like Scalable Wireless Ad hoc Network Simulator (SWANS) [9] or VANET Mobility Simulation (VANetMobiSim) [10] also aim at simulating V2V or VANET communication and are based on an interaction or cooperation of two simulations. In this kind of collaboration, one tool is responsible for the communication part of the simulation and one or more tools are responsible for modeling the traffic. The communication between the systems is done via Transmission Control Protocol (TCP), which allows for the different simulation parts to be on either the same computer or installed on different computers connected through the internet. Such a system was implemented in [11] by using Unity 3D for the simulation of the environment and SUMO for simulating the vehicle movement. The communication was based on the Traffic Control Interface (TraCI) provided by SUMO.

An additional approach which aims at linking two different

 DOI: 10.1109/IVS.2017.7995845

simulation tools was presented in [12]. It focused on generating a tool for simulation of Vehicle-to-everything (V2X) communication based on the combination of the simulation systems SUMO and VISSIM [13]. The authors focused in particular on investigating issues regarding performance and simulation accuracy.
Similar to the approach presented in this work, although based on different technologies, was that of [14]. Inter-vehicle communication was taken into account to develop a platform that tested novel, innovative vehicular applications, including a driver's perspective of the VANET environment. Driver behavior is reflected in the VANET simulation system, affecting the mobility of the cars in the vicinity and providing the intelligent driving model with new realistic features.
Another work related to communication between two or more simulators was presented in [15]. One simulation component was responsible for the vehicle and environment, while the second simulation was responsible for the VANET communication. The results of this work showed a bottleneck-effect in the TCP communication between the two simulations when a high amount of vehicles or objects were simulated.
While there is a wide range of resources and information about implementation of VANET-based simulators, most of them focus on the evaluation of different communication protocols and their performance. Evaluation of new in-vehicle technology that bases on simulation environments with VANET capabilities can be only found in few works (e.g. [16]). The lack of tools to perform specific research on VANET-based ADAS motivated this work. We present the design and development of a flexible, low cost tailored simulation tool to the specific requirements for investigating the effect of V2V communication on driver's response on the basis of an ADAS for safety distance.

III. Simulation Platform Implementation

After considering several technologies and approaches for developing the simulation platform, we decided to build it on top of the Unity game engine due to its modularity, scalability, mature scripting, and physics engine, among other characteristics. We then investigated the potential combination with different simulator platforms (e.g. Omnet++ with SUMO) and determined that a simulation based on both tools will not run until the network calculation is finished, and therefore, we found it unsuitable for combination with Unity 3D. We simulate the V2V communication within Unity 3D to maintain control over what is calculated, in which frame it appears and at which moment it is displayed to the user.
The resulting Unity 3D-based implementation of the VANET communication simulation (coded in C#) is able to calculate the communication between the different vehicles while taking into account bad reception resulting from long distances and/or building interference. The V2V communication was established relying on the Self-Organised Time Division Multiple Access (SOTDMA) [17] protocol, the real-time calculation and the direct integration of the data transmission in the driving simulation. The platform consists of three components that are responsible for the functions below.

A. Simulation setup

The simulation setup contains the base functions for the generation of the road network, importing the provided OSM data and generating the 3D environment. Functions in this component are also responsible for the generation and initialization of all vehicles and provide the necessary structure for the VANET communication. Simulated vehicles (SCV) create realistic traffic while also acting as transmitter and receptor stations for the V2V communication. The ego vehicle is controlled by the user. We labeled it User Controlled Vehicle (UCV).

B. Simulation execution

This component contains the functions that are used to execute the simulation. For example, functions which are responsible for managing user input from a keyboard or a steering wheel and functions related to the simulation of driving or the communication behavior of the vehicles. This component also contains the functions for the simulation of ADAS.

C. Data logging

The functions responsible for the collection and storage of situations and events that occur during the runtime of the simulation belong to the data logging component. These functions are necessary to allow the analysis and comparison of driving behaviors and provide data for the evaluation of safety-related technology.

IV. Driving Map and Environment Implementation

To achieve a balance between realism and performance, we designed the simulation environment based on the following road network elements:

A. Road network and buildings creation

- Road type: to identify and parse nearly all highway tags used in OSM, from motorways to cycling paths. Due to the focus of the simulation on motorized traffic, the current simulation does not parse and generate pedestrian roads and small paths.
- Lane number and turn lanes: the platform evaluates both the lane number and the turn lane information of each road segment in the data and generates a similar road segment in the simulation.
- Traffic signs: the map generation function generates traffic signs based on two main sources in the data. First are the explicit positioned traffic signs, which include "yield" and "stop" signs as well as traffic lights. Implicit traffic signs are also generated, including speed limits which are not placed as nodes in the map data but rather are added directly to the related road segment. Here the simulator looks for locations in the road network where two different speed limits share the same node. At this node a sign is generated for both speed limits. Based on this approach the simulation also creates signs for implicit speed limits which are announced through the road type or city-limit signs.

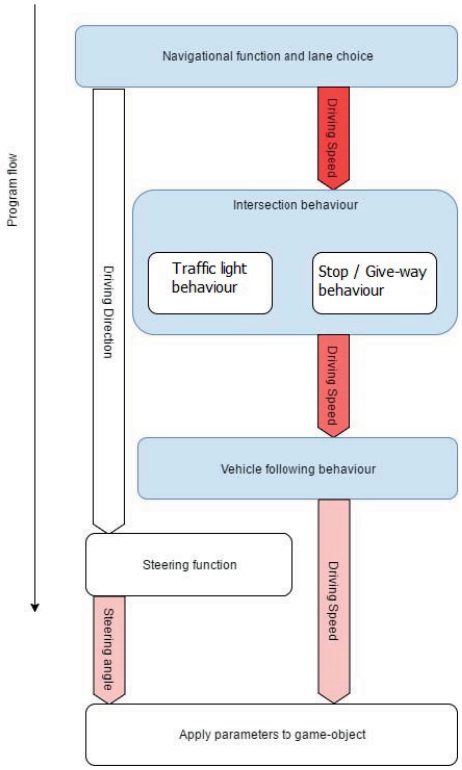

Fig. 1. Model structure used in the simulation to implement different driving behaviors. The figure also shows how the driving parameters are adapted during the behavior analysis.

- One-way roads: just as with the speed limit, one-way roads are tagged on the road segment and not at the position of the sign. The simulation determines the correct sign for the intersection, placing it between the two-way and one-way roads.

B. Behavior of vicinity vehicles

Regarding vehicle dynamics each individual vehicle offers a wide range of attributes for distribution of driving parameters which can be set at the time of the generation (e.g. maximum acceleration and deceleration, weight and wheel friction). The movement and behavior of the simulation-controlled vehicles is based on a multistage model as shown in Figure 1.

The navigation function generates the ideal movement behavior and certain functions, modify the provided movement parameters to the current road situation. Once the function provides general driving direction, allowed maximum speed and distance to the next waypoint node, then the first behavior function initializes and verifies if the vehicle is approaching an intersection, and if so, whether a traffic light or a stop sign exists. The driving speed is determined according to whether the driver needs to turn at a specific intersection and whether a traffic light or stop sign exists. If the current traffic light is red, the vehicle stops 6m before the intersection and waits until it changes to green. At a stop sign the vehicle pauses for at least two frames and verifies whether there are approaching vehicles within 200m. In affirmative cases, the vehicle waits, otherwise it continues its route. At a give-way sign the vehicle reduces its speed to 5km/h and verifies whether there are approaching vehicles within 200m. In affirmative cases, the vehicle stops, otherwise it continues its route. If there is no intersection the driving speed is not adjusted.
After the intersection behavior, the next function establishes whether there are other vehicles in the near vicinity that would have an influence on the first vehicle's movement due to differences in current speed and acceleration. If another vehicle is detected ahead, the SCV sustains a 2-second distance from the other vehicle by comparing the speeds of both vehicles and reducing its own speed by 80% if necessary. In the event that the velocity is too high in relation to the calculated forces from the physics engine, driving speed is reduced to prevent the vehicle from leaving the road.

V. Vehicle Generation and Routing Approach

A. Vehicle generation

Generally speaking, in order to create realistic vehicle movement based on a potentially changing road network, it is common to use some sort of generation process where vehicles are created which follow a pre-defined route toward a removal point. This approach requires a defined process to determine the generation point placement that is directly related to the size of the road network. In our simulation vehicles are only created at road nodes of the following OSM road types: primary, secondary, tertiary and unclassified roads, motorways and residential roads. The generating node is located at an angle of $+/-45°$ ahead of the vehicle. Once the node on which the vehicle will be generated is defined, a second algorithm generates a random destination node and sends the two node identifications to the routing algorithm. This then returns a waypoint path consisting of each road node from the origin of the vehicle to the defined destination. For navigating and movement of the SCV, this waypoint path is the key element. It is characterized mainly by an uninterrupted chain of nodes from the origin to the destination, wherein the connecting roads between the nodes always form a straight line. For a vehicle to navigate the road network, it has to target the direction of the next waypoint. The flowchart in Figure 2 shows the general algorithm of the navigation function and its hierarchical elements. It determines if the vehicle has reached its destination, the right lane in which to drive and the correct speed.

B. Routing approach

The applied routing algorithm A* uses heuristics and a weight function to determine the shortest path between two nodes in the road network. This algorithm is considered one of the fastest routing algorithms developed; however, the computational power it requires for calculation is quite large. To reduce this burden the driving simulation uses a combination of two common approaches. The first approach is done mainly by low-cost navigation devices that use few computational resources, or which are used in use cases where the road network does not change. This method is called pre-calculated routing, because the best route between

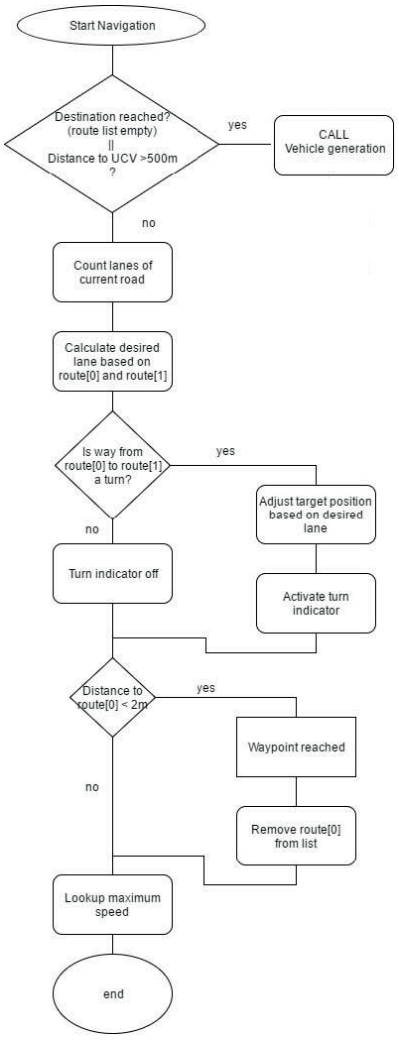

Fig. 2. Algorithm for the navigation function showing the hierarchy of the functions

two points is calculated in advance. It has considerable advantages in terms of decreasing the routing calculation time if the same route is asked by multiple users. The second approach calculates the route when it is needed and is applied mainly in navigation devices for in-vehicle use. A certain time is required to calculate the route in real-time, this resulting in a reduction of necessary storage capacity on the device. In our simulation we rely on a combination of both approaches: the simulation calculates the route in real time, while also storing the result in a file. This additional storage process results in a routing table for the road network. The routing function evaluates whether the route already exists in the file, an evaluation which uses fewer resources than calculating each route anew. This approach with a routing table is the fastest and was therefore used in the defined scenario.

VI. Implementation of the Vehicular Ad Hoc Network communication

The simulation of the V2V communication in the driving simulator is based on the SOTDMA protocol and the implementation of the Cooperative Awareness Message (CAM) type. Some characteristics of the used protocols are modified for a better fit with the structure of the simulator. For example, communicating nodes were initialized independently of the protocol to decrease the initialization time for new communication nodes in the simulation and reduce computing needs improving efficiency. The structure for the V2V communication is illustrated in Figure 3. The VANET communication is based on the following objects:

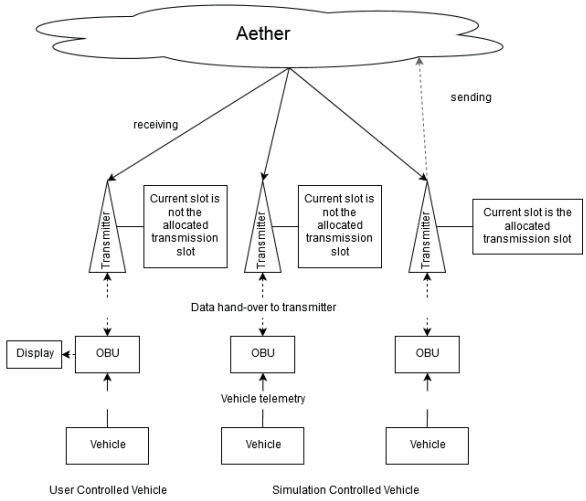

Fig. 3. General structure of the implemented V2V communication in Unity 3D

A. Aether

This game object simulates the concept of the signal distribution. In the simulation the game object is the common interface and simulates the air and geographical situation of the simulation area. Each on-board unit (OBU) sends its message to the aether element. The script in the game object evaluates which vehicles are able to receive the message based on an algorithm which calculates the transmission shadowing of buildings and the maximum transmission range of the simulated equipment. The aether script is also responsible for simulating problems related to hidden nodes by transmitting damaged messages to vehicles which would receive messages from different transmission nodes at the same time slot. If two vehicles try to transmit in the same time slot, the aether object transmits the messages only to vehicles which are not in the overlapping interference area. In this area, depending on the possible adjusted transmission behavior related to shadowing and transmission range, messages are not recognizable.

B. Transmitter

The transmitter simulates the antenna of the vehicle, which sends and receives the message packages to the aether.

C. On board unit

The OBU game object simulates the board computer for the V2V communication. It collects the telemetry data from the vehicle necessary for the generation of the CAM package, generates the data and sends it to the transponder

game object. This object is also responsible for the parsing, interpretation and display of the incoming CAM packages. For this it evaluates the incoming data and calculates the position of the vehicle which transmitted its data into the current transmission slot. It additionally creates an internal map conveying the distances, driving directions and speeds of all vehicles that received a V2V message in the last five frames. Based on this map the script evaluates if a vehicle matches the following criteria: a) vehicle is in close proximity to the receptor vehicle; b) vehicle drives in the same direction as the receptor vehicle; c) vehicle is in front of the receptor vehicle; d) vehicle uses the same lane as the receptor vehicle; e) acceleration of the transmitting vehicle is smaller than the acceleration of the receptor; f) calculated time gap between the vehicles is below two seconds; g) emergency breaking bit is not set in the transmitting vehicle. If a match is identified, the object will display to the user safety distance information that informs them about a potential danger.

D. Message distribution through the aether object

As previously mentioned, the aether object is responsible for the distribution of the CAM packages to all transmitter objects currently active in the simulation. For this distribution, the script within the aether object is combined with the procedure which evaluates whether the object vehicle is able to receive the message. For the distribution of messages to different vehicles the script uses the LateUpdate() event function of Unity. It guarantees that the function is executed after all transmitter objects are able to receive and send their data. To this end our algorithm verifies if a message was transmitted in the current time slot, and it starts to evaluate the shadowing and transmission range restrictions for the message. All messages are distributed into one time slot after they were created. The transmission time was approximately 0.05 seconds.

E. Algorithm to determine building shadowing and transmission range

To determine if the message broadcasted by a certain vehicle could be received by another vehicle, we relied on a building shadowing- and transmission-range calculation. In signal transmission shadowing occurs when signals are blocked due to objects or other reflective surfaces. We developed an algorithm that compares all receptor vehicles to the information provided by the transmitted information. It verifies the transmitters range to determine if a vehicle is able to receive the message, and uses the native Unity 3D function Physics.Linecast(Vector3 position1, Vector3 position2) to determine if there is a straight line of sight between the receiving and transmitting position. This function generates a straight line between the two given Vector3 coordinates and evaluates whether the line is intersected by a solid game object (determined by its rigid body component). If there is no intersection, the function returns “true”, otherwise it returns “false”. This value is then used in the code to evaluate the blocking. Unfortunately the Linecast function is not able to determine if the line of sight is blocked by a vehicle or a building, which means it is necessary to prevent vehicles from being in the line of sight. To solve this problem we set the y position of the start point of the Linecast function 10 meters above the sending vehicle. This shadowing through vehicles is quite improbable and only happens in between long distances when the angle between the line of sight and the plane becomes small. The differentiation between the ego vehicle and the SCV is necessary because of the differing components of the game objects, which make it impossible to put them into the same game object array.

VII. Validation of the Developed Simulation Platform

To validate the adequacy of the developed simulation platform for assessing driving assistance systems with and without VANET capabilities we implemented two ADAS as follows:

A. Rear-mounted distance warning system

The rear-mounted distance warning system [1] uses two low-cost cameras to create a stereoscopic image from the rear window of the vehicle which allows the calculation of the distance between the leading and following vehicles. The combination of this distance with the current vehicle’s speed is then used to evaluate if the distance between the two vehicles is sufficient to guarantee safety. If the distance is too small, the driver of the following vehicle is informed by means of a display located in the rear end of the leading vehicle, which shows a short “keep distance” text message.

B. Vehicle-to-vehicle based distance warning system

The implemented distance warning system is based on the communication between two vehicles and takes advantage of defined information provided by the CAM data frames, which allows the extraction of all necessary information to detect the distance between two vehicles. The VANET system displayed to the driver the time-gap between the ego and the leading vehicle, and if this time-gap was less than three seconds a warning was issued within the ego vehicle until the driver adjusted the gap to the recommended two seconds. For the implementation of a distance warning system or collision avoidance system it is necessary to gather the following information: 1) position of the own vehicle, 2) position of connected vehicle, 3) driving direction, 4) current driving lane for both vehicles, 5) speed of the own vehicle, 6) speed of the connected vehicle, 7) relative acceleration between the two vehicles. The first two elements are necessary to calculate the distance between the two vehicles, but this information is not enough to evaluate if the distance between the two vehicles is sufficient or not. Elements 3 and 4 are used to determine if the two vehicles are following each other. The last 3 points are necessary to determine if the distance is sufficient at the current speed and if the time gap between the two vehicles is changing.

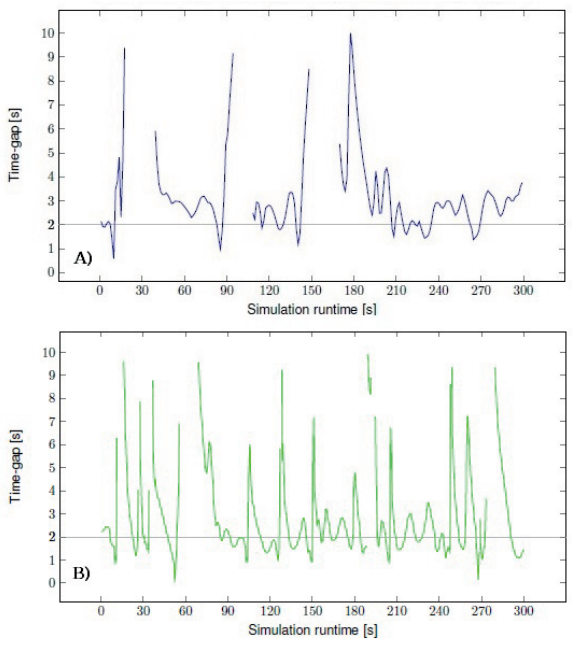

Fig. 4. Time-gap for participant 3 using the following systems: A) Rear-mounted distance warning system and B) Vehicle-to-vehicle based distance warning system

C. Experiment implementation and results

We performed several tests with 4 persons (2 males, 2 females, mean age = 34, SD = 1.5), 3 daily commuters, 1 occasional driver and compared the systems presented above. The test persons had to drive 3 sessions within 15 minutes through a predefined path without secondary tasks in a medium to high traffic density scenario. During the first 5 minutes no systems were activated (baseline). During the second and third 5-minute periods the systems described above were activated alternatively. The participants were instructed to follow a leading vehicle respecting the traffic rules.

Results showed that both systems affected driver response in the simulated environments. Figure 4 depicts the results from one participant as an example. Drivers interacting with the rear-mounted distance warning system reacted to warnings by suddenly braking and increasing the distance to the leading vehicle, while the drivers interacting with the VANET-based system adjusted their driving regularly according to the time-gap displayed in the vehicle. Which system provided a higher level of safety was not determined in this work.

VIII. CONCLUSION AND FUTURE WORK

We presented in this paper a simulation platform to assess driving assistance systems with and without Vehicle to Vehicle communication capabilities. Results regarding the proof of concept of the simulator to test new in-vehicle technology validated the presented application in its current state. In future work we will optimize the road network by defining curves as a single road segment, therefore simplifying the calculation. While the current implementation is based mainly on the line of sight information to determine if a transmission is blocked or not, it is possible to implement a more sophisticated algorithm to improve the shadowing in the future. Interference in the communication, for example in the case of being two vehicles transmitting in the same time slot, can also be addressed through an improvement of the transmission behavior based on detailed signal to noise ratio.

ACKNOWLEDGMENTS

This work was supported by the KiTSmart Project - City of Vienna Competence Team for Intelligent Technologies in Smart Cities, funded by national funds through the MA 23, Urban Administration for Economy, Work and Statistics, Vienna, Austria and the Photonics Project: Foundations and industrial applications.

REFERENCES

[1] C. Olaverri-Monreal, R. Lorenz, F. Michaeler, G. Krizek, and M. Pichler, "Tailigator: Cooperative system for safety distance observance," in *Proceedings 2016 International Conference on Collaboration Technologies and Systems, Orlando, Florida, USA*. IEEE, 2016, pp. 392–397.

[2] J. Gonçalves, R. J. Rossetti, and C. Olaverri-Monreal, "IC-DEEP: A serious games based application to assess the ergonomics of in-vehicle information systems," in *2012 15th International IEEE Conference on Intelligent Transportation Systems*. IEEE, 2012, pp. 1809–1814.

[3] M. Bröckerhoff, "Pelops whitepaper," *Forschungsgesellschaft Kraftfahrwesen mbH Aachen (FKA), Aachen, Germany, Tech. Rep.*, 2010.

[4] A. Richter and H. Friedl, "Parametrisierte Stadtmodelle für Fahrsimulatoren," http://elib.dlr.de/96485/, 21.05.2015, (accessed on: 22.02.2016).

[5] N. Sparwasser, M. Stöbe, H. Friedl, T. Krauß, and R. Meisner, "SimWorld – Automatic Generation of realistic Landscape models for Real Time Simulation Environments – a remote sensing and GIS-data based processing chain," 2007, (accessed on: 26.08.2016).

[6] T. Willke, P. Tientrakool, and N. Maxemchuk, "A survey of inter-vehicle communication protocols and their applications," *IEEE Communications Surveys & Tutorials*, vol. 11, no. 2, pp. 3–20, 2009, 10.1109/SURV.2009.090202.

[7] "Ns-3," https://www.nsnam.org//.

[8] Omnet, "Omnet++ discrete event simulator - home," https://omnetpp.org/, (accessed on: 25.07.2016).

[9] "Swans- scalable wireless ad hoc network simulator user guide," http://jist.ece.cornell.edu/swans-user/index.html//.

[10] "Vanetmobisim," http://vanet.eurecom.fr///.

[11] C. Biurrun-Quel, L. Serrano-Arriezu, and C. Olaverri-Monreal, "Microscopic driver-centric simulator: Linking Unity3d and SUMO," in *Recent Advances in Information Systems and Technologies. Proceedings 5th World Conference on Information Systems and Technologies (WorldCist'17), Porto Santo Island, Madeira, Portugal*. Springer, 2017.

[12] D. Rieck, B. Schünemann, I. Radusch, and C. Meinel, "Efficient traffic simulator coupling in a distributed v2x simulation environment," in *3rd International ICST Conference on Simulation Tools and Techniques*, G. Stea and L. F. Perrone, Eds.

[13] "PTV Vissim," http://vision-traffic.ptvgroup.com/de/home/.

[14] P. Gomes, C. Olaverri-Monreal, M. Ferreira, and L. Damas, "Driver-centric vanet simulation," in *International Workshop on Communication Technologies for Vehicles*. Springer, 2011, pp. 143–154.

[15] S. Guan, R. E. D. Grande, and A. Boukerche, "Real-time 3d visualization for distributed simulations of vanets," in *2014 IEEE/ACM 18th International Symposium on Distributed Simulation and Real Time Applications (DS-RT)*, pp. 138–146.

[16] C. Olaverri-Monreal, P. Gomes, R. Fernandes, F. Vieira, and M. Ferreira, "The see-through system: A vanet-enabled assistant for overtaking maneuvers," in *Intelligent Vehicles Symposium (IV), 2010 IEEE*. IEEE, 2010, pp. 123–128.

[17] K. Bilstrup, E. Uhlemann, E. Ström, and U. Bilstrup, "On the ability of the 802.11 p mac method and stdma to support real-time vehicle-to-vehicle communication," *EURASIP Journal on Wireless Communications and Networking*, vol. 2009, no. 1, p. 1, 2009.