2021 IEEE Intelligent Vehicles Symposium (IV)
July 11-17, 2021. Nagoya, Japan

Validation Method of a Self-Driving Architecture for Unexpected Pedestrian Scenario in CARLA Simulator

Rodrigo Gutiérrez[1], J. Felipe Arango[1], Carlos Gomez-Huélamo[1], Luis M. Bergasa[1], Rafael Barea[1] and Javier Araluce[1]

***Abstract*— This paper introduces a method to validate autonomous navigation frameworks, in simulation using CARLA Simulator, fulfilling the requirements of the Euro-NCAP evaluation. We propose the protocol for evaluating an unexpected pedestrian scenario, where a walker suddenly invades the road and the vehicle has to react in a safe way. Standard validation metrics are created for this use case, which are generalizables for other use cases. To support the proposal, we describe our ROS (Robot Operating System) based Self-Driving architecture, open source and implemented in an electric vehicle. Then, we explain the procedures and requirements needed for the validation protocol that we propose. Finally, we show the metrics and results obtained in simulation for different ego-vehicle velocities and weather conditions. The scenarios implemented in Carla are publicly available [2].**

I. INTRODUCTION

Autonomous vehicles (AVs) have become one of the most important engineering challenge of the last decades. These AVs must perform driving behaviours with a greater reliability than humans, solving transportation problems such as traffic jams or accidents. Many agents that can influence in drivers decisions makes the driving task really complex, that is why multiple sensors perceiving the surrounding environment in real-time and robust algorithms evaluating the obtained data are needed for this purpose.

In recent years, AVs have been developed for both research and commercial objectives. As result of this competition, great self-navigation systems have come up. Some of them in the automotive industry such as Tesla, Google or Uber [1], and others in competitions such as DARPA Grand Challenge [2] or CARLA Autonomous Driving Challenge [3].

We are a research group from the University of Alcalá (Spain). In our project, Techs4AgeCar, the main goal is to implement an open source autonomous electric car [4]. For this purpose, we undertake all the design stages involved in the design of an autonomous vehicle: mechanics, electronics, perception, planning, control, navigation, validation in simulation and in a real environment.

In this paper we present our architecture, implemented in an electric vehicle. We use ROS [5] as an standard of communication for our software modules, which are encapsulated in a Docker container [6] that can be run in both real environment and simulation, using the novel and hyper-realistic simulator CARLA [7]. Each of these modules is related with a navigation layer: decision-making, control, localization, mapping, planning and perception [8].

Our goal in this paper is to validate our architecture by designing and implementing standard protocols. In order to do this, we have developed detailed tests and assessments procedures using standard metrics. These test scenarios were set up based on Euro-NCAP [9]. From all of them, we will focus on the unexpected pedestrian scenario, where the vehicle has to react safely against a sudden road invasion by a pedestrian. We have created this scenario for different velocities and weather conditions. This paper describes both the test design process and the assessment protocol.

Fig. 1: Scenario simulation overview

The remaining content of this work is organized as follows. The next section presents the main vehicle testing protocols of the state of the art and the CARLA Autonomous Driving Leaderboard. Section 3 briefly describes our vehicle architecture. Section 4 explains the validation protocol for the unexpected pedestrian scenario in detail. Section 5 shows the obtained results. Section 6 deals with the conclusions of the paper and with the future works.

II. RELATED WORKS

A considerable amount of research works and studies, related to pedestrian detection and collision avoidance behavior have been presented in the literature [10], [11], where perception or control module are analyzed in depth. However, our goal in this work is to evaluate a whole architecture, where all modules are integrated, using common metrics for all frameworks. In this context, we can find the following options:

*This work was funded in part from the Spanish MICINN/FEDER through the Techs4AgeCar project (RTI2018-099263-B-C21) and from the RoboCity2030-DIH-CM project (P2018/NMT- 4331)), funded by Programas de actividades I+D (CAM) and cofunded by EU Structural Funds.

[1]R. Gutiérrez, J.F. Arango, C. Gomez-Huélamo, L.M. Bergasa, R. Barea and J. Araluce are with the Electronics Dpt., University of Alcalá (UAH), Spain. {rodrigo.gutierrez, juanfelipe.arango, carlos.gomezh, luism.bergasa, rafael.barea, javier.araluce}@uah.es

[2] https://github.com/RobeSafe-UAH/scenarios

2021 IEEE Intelligent Vehicles Symposium (IV) | 978-1-7281-5394-0/21/$31.00 ©2021 IEEE | DOI: 10.1109/IV48863.2021.9575884

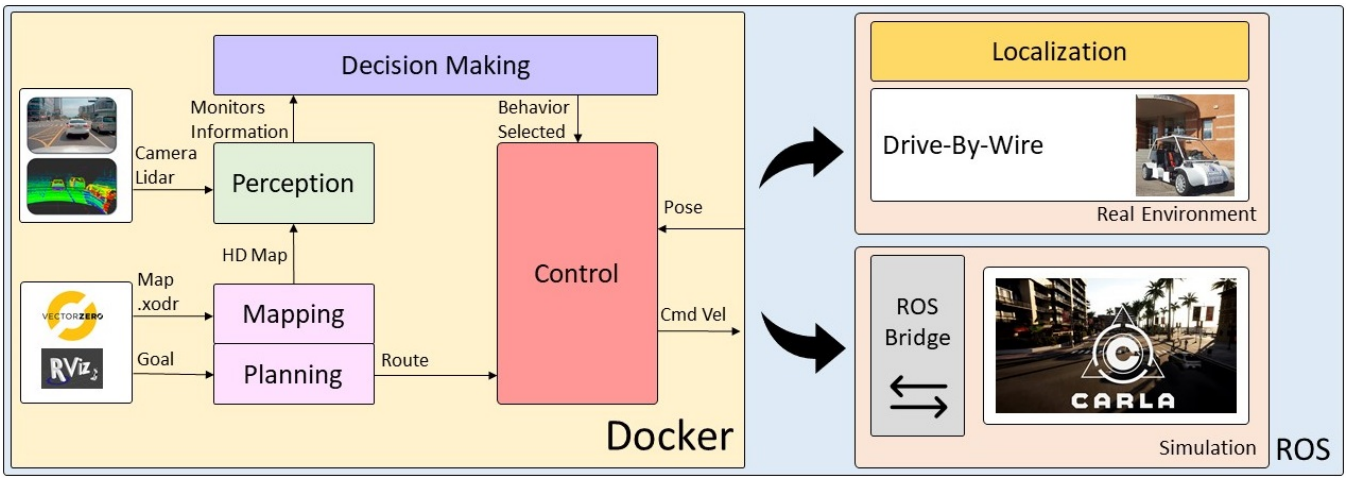


Fig. 2: Techs4AgeCar architecture

A. New Car Assessment Programmes

There are several New Car Assessment Programs (NCAPs), which are a set of protocols to evaluate the safety of vehicles. Currently, evaluations are focused on the structure of vehicle and the Advanced Driver Assistance Systems (ADAS), such as: Adult Occupant Protection (AOP), Child Occupant Protection (COP), Autonomous Emergency Braking (AEB) , Speed Assist Systems (SAS), etc. The most important NCAPs, at the present time, are:

- European New Car Assessment Program (Euro-NCAP): is the most widely used performance assessment, was founded in 1997, within the scope of the collaboration of the countries of the European Union [12].
- China New Car Assessment Program (C-NCAP): is a research and development benchmark for vehicle manufacturers in Asio. Most of its program is based on the Euro-NCAP [13].
- National Highway Transportation Safety Administration (NHTSA): is an agency of the U.S. federal government, part of the Department of Transportation. They have published research reports, guidance documents, and regulations on vehicles equipped with ADAS [14].
- Autoreview Car Assessment Program (ARCAP): is the first independent evaluation program for cars in Russia. They publish their studies in a newspaper called Autoreview [15].

But these programs, so far, do not have specific protocols to evaluate self-driving architectures. Also, they are quite different among them, they use different scenarios, parameters, ratios, etc. They also do not specify the equipment to be used.

B. CARLA Autonomous Driving Leaderboard

The objective of the CARLA Leaderboard is to evaluate the performance of different navigation architectures in realistic traffic situations. It is an open-source platform that allows square and reproducible evaluations.

The autonomous navigation architectures are confronted by a set of predefined routes. For each route, vehicles start at a predefined point and must navigate to a destination point. The routes will occur in a variety of areas, including highways, urban scenes and residential districts.

The multiple traffic situations are based on the NHTSA typology. The simulation weather can be changed, allowing evaluation in a variety of weather conditions (daylight, sunset, rain, fog, and night).

At the time of writing this article, there are no standard evaluation protocols for autonomous vehicles, which allow us knowing the performance of each architecture for different scenarios. On the other hand, the CARLA Leaderboard is not easily reproducible in a real environment.

In this paper, we provide a protocol to validate autonomous navigation architectures, in simulation and in real scenarios, giving a global score about its performance. We also provide the results obtained for our architecture, which will serve as a base to compare the performance of different architectures.

III. Our autonomous architecture

Our autonomous navigation architecture is divided in different modules, as shown in Fig. 2. These modules process the information in an asynchronous way and share information using ROS inter-process communication, allowing the different modules being executed in parallel. The subscribing/publishing method allows communication without loosing information nor blocking messages between modules. In this paper we focus in the modules that play an important role in the unexpected pedestrian scenario:

A. Mapping/Planning module

One of our framework strengths is its portability between simulation and real environment. A map based on the university campus has been designed using the VectorZero RoadRunner [16] application. The map information is processed using the CARLA Python API [17], in order to generate the user interface in RVIZ.

For the path planning task, some code has been developed in combination with the ”libcarla” library to create a graph describing the road map. A destination goal is introduced by

the user, then the path planning has been solved applying the graph-based method A* [18].

A list of waypoints describe this trajectory, which is published in a ROS topic. Also, there is an online planning module, that provides information about the lanes state and the regulatory elements existing in the HD map to the perception module.

B. Perception module

In terms of environment perception, we carry out tracking-by-detection [19] using as input Bird's Eye View objects obtained through the sensor fusion of depth map and stereo RGB camera information. Despite the fact that in our real-world prototype we perform a late sensor fusion between the BEV stereo camera objects and the BEV laser objects branch, at the moment of writing this article CARLA does not offer enough quality in its laser pointcloud, even increasing the number of channels and points per second, so as to obtain similar results that in real-world datasets. Then, this set of BEV obstacles feeds a simple yet accurate tracking pipeline made up by a BEV Kalman Filter [20], that predicts the state of the objects from the current frame and update the object state based on the detected bounding boxes at current frame, and a Khun-Munkres (a.k.a Hungarian algorithm [21]) that associates the predicted trajectories with current detections. Finally, a B/D (Birth and Death) memory deals with the newly appeared trajectories (those matched trajectories that exceeds f_{min} frames) and dissapeared trajectories (unmatched trajectories exceeding age_{max} frames).

C. Decision Making module

The decision making module evaluates the information provided by the perception module, the actual state of the vehicle and the HD map features to select the adequate behaviour for each driving situation. We define two types of behaviours:

Background behaviours, just as the unexpected pedestrian or the adaptive cruise control (ACC). These are continuously running, expecting a close vehicle or pedestrian to reduce the vehicle speed in a safe way. Standard use cases, as might be a stop, a give way, an overtaking or a crosswalk. This module is done using Petri Nets. For more information we refer the reader to our previous publication [3].

D. Control module

Our controller [22] performs a smooth interpolation of the waypoints given by the planner. Before the navigation starts, a velocity profile is generated using the curvature radius of each trajectory section. During the navigation the speed command is adjusted using this profile and the steer command is set using LQR techniques to ensure the trajectory tracking. The existing delays in the localization module and the vehicle actuators is compensated in the control loop.

E. CARLA Simulator

CARLA is an hyper-realistic simulator developed to design and validate autonomous driving systems. This simulator provides open digital features that can be used to create scenarios, vary weather conditions, define different sensors distributions, generate maps and much more. We aim to meet an architecture where the results in the simulator can be extrapolated to the real environment, so we can develop algorithms and validate them before we test them in our real vehicle.

We use the provided carla-ros-bridge as an interface between the simulator and our architecture. All sensors data of the simulator is published in ROS topics by this package, also the output signals generated by our system is commanded to the simulated vehicle through this bridge.

F. Localization

This module is in charge of positioning the vehicle on a map with centimeter precision and in real time. Accurate and robust localization is a primary task for autonomous vehicle navigation.

In a real environment, we estimate the vehicle's pose using the fusion of data provided by a differential GNSS (Global Navigation Satellite System) and by an encoder-based odometry. To improve the accuracy and reliability of the GNSS system, differential positioning techniques (DGNSS) and Real Time Kinematics (RTK) were used [23].

On the other hand, in simulation, CARLA provides the exact location of the vehicle via ROS, so a localization module is not required.

G. Drive-By-Wire

The Drive-By-Wire module, which is implemented in the real vehicle, receives the steer and speed commands sent by the Control module and generates the electric signals to feed the electric motor and the steering wheel. [24].

We have implemented PI controller that equals this module in simulation. This module receives the controller output and generates normalized throttle, brake and steer commands, which are the input signals required by the simulated vehicle.

IV. Test Validation Protocol

We propose a test validation protocol with standard metrics for autonomous vehicles. For the unexpected pedestrian use case, we used the Euro-NCAP AEB VRU test protocol v3.0.2. [25] as reference, adjusting it for an autonomous navigation architecture.

A. Test Scenarios

We implemented two of the six scenarios proposed by Euro-NCAP. Fig. 3 shows Car to Pedestrian Nearside Adult (CPNA) scenario. The pedestrians starts in the closest sidewalk to the vehicle.

Fig. 4 shows Car to Pedestrian Farside Adult (CPFA) scenario. The pedestrian starts in the farthest sidewalk to the vehicle. In both cases, the pedestrian invades the path unexpectedly and the vehicle must detect the pedestrian

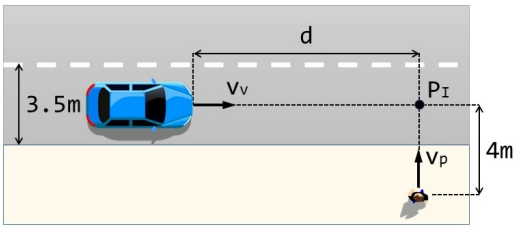


Fig. 3: Car to Pedrestian Nearside Adult (CPNA) scenario

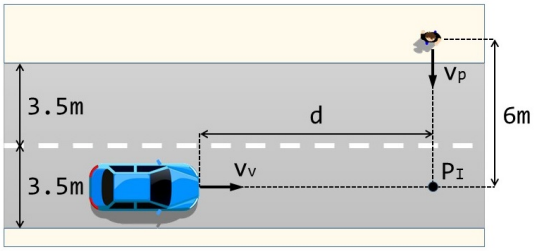


Fig. 4: Car to Pedrestian Farside Adult (CPFA) scenario

and react quickly, avoiding the collision or reducing the impact speed as much as possible. Avoidance maneuver is not contemplated in the protocol.

The pedestrian speed (v_p) is 5 km/h for the CPNA scenario and 8 km/h for the CPFA. The vehicle speed (v_v) ranges from 10 km/h to 60 km/h. We calculate the distance (d), at which the pedestrian must start walking so that impact point (P_I) is in the center of the lane. Table I shows the parameters for each test.

TABLE I: Test parameters

Scenario	v_v	v_p	d	$score_{max}$
CPNA	10 km/h	5 km/h	8.0 m	1.0
	20 km/h		16.0 m	1.0
	30 km/h		24.0 m	2.0
	40 km/h		32.0 m	3.0
	50 km/h		40.0 m	2.0
	60 km/h		48.0 m	1.0
CPFA	10 km/h	8 km/h	7.5 m	1.0
	20 km/h		15.0 m	1.0
	30 km/h		22.5 m	2.0
	40 km/h		30.0 m	3.0
	50 km/h		37.5 m	2.0
	60 km/h		45.0 m	1.0

B. Pedestrian Target

A 3D human appearance dummy is required for the scenario. It has to be flexible and manageable in order to acquire a walking position, and has to be designed so cameras, LIDARs and radars can detect it. The adult dummy height is 180 cm and its clothing is made up by a long-sleeve black shirt and blue trousers as shown in Fig. 1.

C. The Score

We propose to use standard metrics to validate different self-driving architectures and to can compare them in an easy way. Using these metrics we calculate an score, that evaluates the performance of the architecture. We base our score in the Assessment Protocol - VRU v10.0.3 [26]. We generalize the referred protocol, defined only for day and night time, to different weather conditions.

The score for each test is calculated based on the speed reduction of the vehicle:

- For v_v less than or equal to 40km/h:
  - If the vehicle stops without collision, it achieves the highest score:

$$score_{test} = score_{max} \tag{1}$$

  - If the vehicle collides, its score is defined as follows:

$$score_{test} = \frac{v_{test} - v_{impact}}{v_{test}} \cdot score_{max} \tag{2}$$

- For v_v higher than 40km/h:

$$v_{impact} \leq v_{test} - 20 \rightarrow score_{test} = score_{max} \tag{3}$$

$$v_{impact} > v_{test} - 20 \rightarrow score_{test} = 0 \tag{4}$$

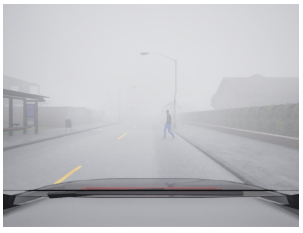
The maximum score ($score_{max}$) for each test is defined in Table I. Each test must be carried out at least three times, and the impact speed (v_{impact}) will be the arithmetic mean of the result obtained in each test. The final score is the arithmetic mean of the scores obtained in each scenario for the different weather conditions.

V. Our environment

(a) Sunny weather

(b) Night-time

(c) Rainy weather



(d) Foggy weather

Fig. 5: Drivers view under different weather conditions

We have elaborated the scenarios described in Section IV using CARLA ScenarioRunner [27], which allows to create complex scenarios. These scenarios have been implemented under different weather conditions, in order to evaluate how environmental factors affect sensors and detection. In Fig. 5

TABLE II: CPNA Results

CPNA									
v_{test}	$score_{max}$	Day		Night		Rain		Fog	
		v_{impact}	$score$	v_{impact}	$score$	v_{impact}	$score$	v_{impact}	$score$
10 km/h	1.00	0.0 km/h	1.00	0.0 km/h	1.00	0.0 km/h	1.00	0.0 km/h	1.00
20 km/h	1.00	0.0 km/h	1.00	0.0 km/h	1.00	0.0 km/h	1.00	0.0 km/h	1.00
30 km/h	2.00	0.0 km/h	2.00	4.62 km/h	1.69	6.74 km/h	1.55	8.36 km/h	1.44
40 km/h	3.00	0.0 km/h	3.00	9.73 km/h	2.27	12.84 km/h	2.04	19.52 km/h	1.54
50 km/h	2.00	19.14 km/h	2.00	27.41 km/h	2.00	35.66 km/h	0.00	49.38 km/h	0.00
60 km/h	1.00	42.93 km/h	0.00	43.67 km/h	0.00	51.37 km/h	0.00	60.04 km/h	0.00
Total	10.00		9.00		7.96		5.59		4.98

we present the simulated CPNA scenario for four weather conditions.

For the implementation of these scenarios we use the OpenScenario standard supported by ScenarioRunner, that provides an execution engine for CARLA. The vehicle, the pedestrian and all the variables that can be modified accomplish the Euro-NCAP requirements. We vary the weather conditions, the pedestrian position and walking velocity. Also the starting walking distance is adequate depending on the vehicle speed.

CARLA provides a bunch of pedestrian textures, but none of them matches the dummy appearance, so we have modified the textures using the Unreal Engine [28] editor in order to reach the specifications detailed in Section IV-B.

VI. Results

Since the Techs4AgeCar project started in 2019 it has been under continuous testing. However, our goal in this work goes further, we aim to validate our architecture in a more detailed way. Therefore, in this section, we present the results obtained under the established protocol.

First, we analyse the temporal response for the CPNA scenario using two graphics, evaluating the architecture for different ego-vehicle velocities (Fig. 6) and for different weather conditions (Fig. 7). Then, we present the results obtained for the CPNA scenario summarized in Table II. Finally, we evaluate our system for CPNA scenario and CPFA scenario by calculating its score in Table III.

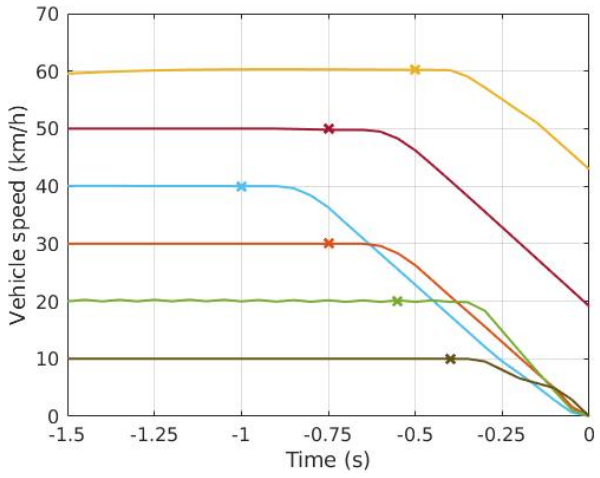


Fig. 6: Time history for sunny weather CPNA scenario

Fig. 6 represents a time diagram of individual tests running different velocities. T_0 corresponds with the moment the vehicle either stops or collides with the pedestrian. The collision is avoided from 10km/h to 40km/h and mitigated for higher velocities. The instant when a predicted collision is detected and a stop command is sent to the vehicle is represented by a cross for each curve.

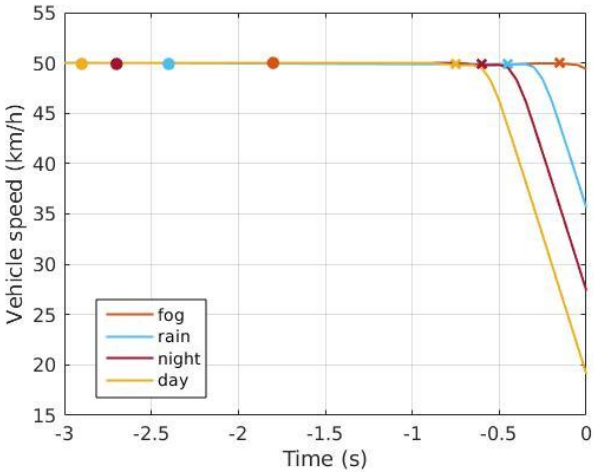


Fig. 7: Time history for different weather conditions at 50 km/h

As we mentioned before, we propose a camera based perception system, expected to have a worse performance being affected by adverse climatic conditions. The tests presented in Fig. 7, where a time diagram of individual tests running under different weather conditions at a velocity of 50km/h is represented, confirm this hypothesis.

The system response is slightly worse at night and with rain; and it barely mitigate the collision with fog. Here, T_0 corresponds with the moment the vehicle collides with the pedestrian. The instant when a pedestrian detection takes place is represented by a dot for each curve and the cross represents the instant when a predicted collision is detected and a stop command is sent to the vehicle.

The test simulation results of the CPNA scenarios are summarized in Table II. As mentioned before, we use a camera based detection method, this explains the slightly worse results while evaluating the night, rain and fog scenarios.

The global score obtained by our autonomous navigation architecture is shown in Table III. The higher pedestrian speed of CPFA scenario in comparison with the CPNA

scenario generates worse results for the former one.

TABLE III: Final Results

Scenario	Day	Night	Rain	Fog	Scenario Score
CPNA	9.00	7.96	5.59	4.98	6.88
CPFA	6.29	5.69	3.96	3.12	4.77
TOTAL	7.65	6.83	4.77	4.05	5.82

VII. Conclusions

A validation method of a self-driving architecture for unexpected pedestrian scenario following the Euro-NCAP standard has been presented in this paper.

The implemented protocol consisted on the study of the scenarios proposed by Euro-NCAP, adapting the specifications to a general navigation architecture, where changes in the weather are considered in the final score. Because of this, we have decided to bring our scenario to CARLA, which allows us to simulate different environments. Results obtained by our architecture for different ego-vehicle speed and weather conditions are exposed and can be used as a baseline for comparison with other frameworks.

We have found some aspects of our architecture that have to be improved by testing it with the proposed scenario. These improvements will help our system to have better performance in more complex autonomous driving environments.

Implementation of new scenarios and additional weather conditions in simulation, as well as test validations for them in simulation and in a real environment will be done in a near future.

References

[1] B. Brown. The social life of autonomous cars. *Computer*, 50(2):92–96, 2017.

[2] Sebastian Thrun. Winning the darpa grand challenge. In Johannes Fürnkranz, Tobias Scheffer, and Myra Spiliopoulou, editors, *Machine Learning: ECML 2006*, pages 4–4, Berlin, Heidelberg, 2006. Springer Berlin Heidelberg.

[3] Carlos Gómez-Huélamo, Javier Del Egido, Luis M. Bergasa, Rafael Barea, Elena López-Guillén, Felipe Arango, Javier Araluce, and Joaquín López. Train here, drive there: Simulating real-world use cases with fully-autonomous driving architecture in carla simulator. In Luis M. Bergasa, Manuel Ocaña, Rafael Barea, Elena López-Guillén, and Pedro Revenga, editors, *Advances in Physical Agents II*, pages 44–59, Cham, 2021. Springer International Publishing.

[4] Projects - robesafe research group. `http://www.robesafe.com/index.php/en/proyects`. Accessed: 2021-02-10.

[5] Morgan Quigley, Ken Conley, Brian Gerkey, Josh Faust, Tully Foote, Jeremy Leibs, Rob Wheeler, and Andrew Y Ng. Ros: an open-source robot operating system. In *ICRA workshop on open source software*, page 5. Kobe, Japan, 2009.

[6] A. A. Mohallel, J. M. Bass, and A. Dehghantaha. Experimenting with docker: Linux container and base os attack surfaces. In *2016 International Conference on Information Society (i-Society)*, pages 17–21, 2016.

[7] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Sergey Levine, Vincent Vanhoucke, and Ken Goldberg, editors, *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78 of *Proceedings of Machine Learning Research*, pages 1–16. PMLR, 13–15 Nov 2017.

[8] C. Gómez-Huelamo, L. M. Bergasa, R. Barea, E. López-Guillén, F. Arango, and P. Sánchez. Simulating use cases for the uah autonomous electric car. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 2305–2311, 2019.

[9] Michiel R. van Ratingen. The euro ncap safety rating. In Alexander Piskun, editor, *Karosseriebautage Hamburg 2017*, pages 11–20, Wiesbaden, 2017. Springer Fachmedien Wiesbaden.

[10] Markus Enzweiler and Dariu M Gavrila. Monocular pedestrian detection: Survey and experiments. *IEEE transactions on pattern analysis and machine intelligence*, 31(12):2179–2195, 2008.

[11] Antonio Brunetti, Domenico Buongiorno, Gianpaolo Francesco Trotta, and Vitoantonio Bevilacqua. Computer vision and deep learning techniques for pedestrian detection and tracking: A survey. *Neurocomputing*, 300:17–33, 2018.

[12] Michiel van Ratingen, Aled Williams, Anders Lie, Andre Seeck, Pierre Castaing, Reinhard Kolke, Guido Adriaenssens, and Andrew Miller. The european new car assessment programme: A historical review. *Chinese Journal of Traumatology*, 19(2):63–69, 2016.

[13] Kuiyuan Guo, Yan Yan, Juan Shi, Runqing Guo, and Yuguang Liu. An investigation into c-ncap aeb system assessment protocol. In *SAE Technical Paper*. SAE International, 09 2017.

[14] Á. Takács, D. A. Drexler, P. Galambos, I. J. Rudas, and T. Haidegger. Assessment and standardization of autonomous vehicles. In *2018 IEEE 22nd International Conference on Intelligent Engineering Systems (INES)*, pages 000185–000192, 2018.

[15] Arcap - autoreview car assessment program. `https://autoreview.ru/arcap/about`. Accessed: 2021-02-10.

[16] Roadrunner user guide - vectorzero support - confluence. `https://tracetransit.atlassian.net/wiki/spaces/VS/pages/740589685/RoadRunner+User+Guide`. Accessed: 2021-02-10.

[17] Python api tutorial - carla simulator. `https://carla.readthedocs.io/en/0.9.2/python_api_tutorial/`. Accessed: 2021-02-10.

[18] Akshay Guruji, Himansh Agarwal, and Deep Parsediya. Time-efficient a* algorithm for robot path planning. *Procedia Technology*, 23:144–149, 12 2016.

[19] Carlos Gómez-Huelamo, Javier del Egido, Luis M Bergasa, Rafael Barea, Manuel Ocaña, Felipe Arango, and Rodrigo Gútierrez. Real-time bird's eye view multi-object tracking system based on fast encoders for object detection. In *2020 IEEE Intelligent Transportation Systems Conference (ITSC)*. IEEE, 2020.

[20] Rudolph Emil Kalman. A new approach to linear filtering and prediction problems. 1960.

[21] H. W. Kuhn and Bryn Yaw. The hungarian method for the assignment problem. *Naval Res. Logist. Quart*, pages 83–97, 1955.

[22] Rodrigo Gutiérrez, Elena López-Guillén, Luis M. Bergasa, Rafael Barea, Óscar Pérez, Carlos Gómez Huélamo, J. Felipe Arango, Javier del Egido, and Joaquin López. A waypoint tracking controller for autonomous road vehicles using ros framework. *Sensors*, 20:4062, 07 2020.

[23] Miguel Tradacete, Álvaro Sáez, Juan Felipe Arango, Carlos Gómez Huélamo, Pedro Revenga, Rafael Barea, Elena López-Guillén, and Luis Miguel Bergasa. Positioning system for an electric autonomous vehicle based on the fusion of multi-gnss rtk and odometry by using an extented kalman filter. In Raquel Fuentetaja Pizán, Ángel García Olaya, Maria Paz Sesmero Lorente, Jose Antonio Iglesias Martínez, and Agapito Ledezma Espino, editors, *Advances in Physical Agents*, pages 16–30, Cham, 2019. Springer International Publishing.

[24] J. Felipe Arango, Luis M. Bergasa, Pedro Revenga, Rafael Barea, Elena López-Guillén, Carlos Gómez-Huélamo, Javier Araluce, and Rodrigo Gutiérrez. Drive-by-wire development process based on ros for an autonomous electric vehicle. *Sensors*, 20:6121, 10 2020.

[25] Euro ncap aeb vru test protocol - v3.0.3. `https://cdn.euroncap.com/media/53153/euro-ncap-aeb-vru-test-protocol-v302.pdf`. Accessed: 2021-02-10.

[26] Euro ncap assessment protocol - vru - v10.0.3. `https://cdn.euroncap.com/media/58230/euro-ncap-assessment-protocol-vru-v1003.pdf`. Accessed: 2021-02-10.

[27] Home - carla scenariorunner. `https://carla-scenariorunner.readthedocs.io/en/latest/`. Accessed: 2021-02-10.

[28] Andrew Sanders. *An introduction to Unreal engine 4*. CRC Press, 2016.