Automatic Traffic Scenario Conversion from OpenSCENARIO to CommonRoad

Yuanfei Lin, Michael Ratzel, and Matthias Althoff

***Abstract*— Scenarios are a crucial element for developing, testing, and verifying autonomous driving systems. However, open-source scenarios are often formulated using different terminologies. This limits their usage across different applications as many scenario representation formats are not directly compatible with each other. To address this problem, we present the first open-source converter from the OpenSCENARIO format to the CommonRoad format, which are two of the most popular scenario formats used in autonomous driving. Our converter employs a simulation tool to execute the dynamic elements defined by OpenSCENARIO. The converter is available at commonroad.in.tum.de and we demonstrate its usefulness by converting publicly available scenarios in the OpenSCENARIO format and evaluating them using CommonRoad tools.**

I. Introduction

Scenarios play a significant role in the development, testing, and validation of autonomous driving systems [1]. However, there is a shortage of both open-source and commonly-used scenarios. Various representation formats for scenarios are supported by different applications, depending on their specific purposes. For example, OpenSCENARIO[1] employs a logical scenario description that consists of a parameterized set of variables. Instead, CommonRoad [2] describes concrete scenarios that are instances of a logical scenario with fixed parameters. For this reason, a converter between different formats is desired to promote the exchange and usability of scenarios. In this work, we present the first openly accessible converter from OpenSCENARIO to CommonRoad, two widely-used formats in the field of autonomous driving [3].

A. Related Work

Next, we review different scenario formats and the works that use them, followed by discussing their capabilities.

a) CommonRoad: CommonRoad scenarios are represented as XML files containing a detailed description of the road network, traffic participant movements, and vehicle planning problems. To facilitate benchmarking of motion planning on roads, CommonRoad provides a range of vehicle models and cost functions. To enable the use of more diverse and realistic scenarios, CommonRoad provides dataset converters[2] to convert real-world data from various sources, such as drones [4]–[9], onboard sensors [10], and infrastructure [11], into a unified representation. One can also create handcrafted or generate safety-critical traffic scenarios based on that data [12]–[14]. In addition, CommonRoad can be coupled with other simulation software platforms such as SUMO [15] and Apollo [16] to test motion planning algorithms in interactive driving environments. The suite of open-source tools provided by CommonRoad is extensive and robust, featuring a drivability checker [17], a set-based predictor [18], a reachability analyzer [19], and a criticality estimator [20]. These tools are designed to be effective and user-friendly in evaluating scenarios, making them a convenient option for various applications. For instance, safe, ethical, and robust motion planning algorithms are benchmarked using CommonRoad scenarios in [21]–[24]. The authors in [25]–[27] utilize CommonRoad tools to demonstrate the game-theoretic aspects of autonomous vehicles. Furthermore, CommonRoad paves the way to use advanced algorithms to facilitate motion planning, such as reinforcement learning [28], [29] and geometric deep learning [30].

b) ASAM OpenX: ASAM[3] is a standardization organization that defines open file formats (aka OpenX) for autonomous driving and traffic simulation. OpenSCENARIO specifies the dynamic aspects of the environment, while OpenDRIVE[4] and OpenCRG[5] define the static elements such as road networks. Additionally, the Open Simulation Interface[6] (OSI) is regarded as a standardized interface that provides easy and straightforward compatibility between different simulation platforms. There exist several tools publicly available for generating OpenSCENARIO and OpenDRIVE files, such as Open Scenario Editor[7], scenariogeneration[8], and MATLAB RoadRunner[9]. For testing and validating autonomous vehicles, OpenSCENARIO can be utilized in simulation platforms like openPASS [31], esmini[10], and CARLA [32] to execute complex traffic scenarios [33]–[36].

c) Other Scenario Formats: GeoScenario [37] is a domain-specific language akin to OpenSCENARIO, designed for constructing test scenarios. Based on perception systems, the authors in [38] develop the tool Scenic that defines scenarios as a distribution over configurations of obstacles. Several additional scenario formats are developed to cater to diverse use cases. Examples of such formats include SceML [39], SDL [40], ADSML [41], Paracosm [42], and MetaScenario [43], among others.

The authors are with the School of Computation, Information and Technology, Technical University of Munich, 85748 Garching, Germany.
{yuanfei.lin, michael.ratzel, althoff}@tum.de

[1] https://www.asam.net/standards/detail/openscenario/
[2] https://commonroad.in.tum.de/tools/dataset-converters
[3] https://www.asam.net
[4] https://www.asam.net/standards/detail/opendrive/
[5] https://www.asam.net/standards/detail/opencrg/
[6] https://www.asam.net/standards/detail/osi/
[7] https://github.com/ebadi/OpenScenarioEditor
[8] https://github.com/pyoscx/scenariogeneration
[9] https://mathworks.com/products/roadrunner.html
[10] https://github.com/esmini/esmini

B. Contributions

In our previous work [44], we developed a map converter from the OpenDRIVE format to lanelets [45]. Lanelets are used by CommonRoad to describe the road geometry. We substantially extend this conversion by encoding the dynamic elements of a scenario specified by OpenSCENARIO. Our converter is expected to be valuable to academic groups and industry professionals alike, given the vast number of openly accessible scenarios available in the CommonRoad and OpenSCENARIO formats.

The paper is structured as follows: Sec. II provides a detailed explanation of the scenario conversion from OpenSCENARIO to CommonRoad, further elucidated with numerical examples in Sec. III. The conclusion is in Sec. IV.

II. Conversion from OpenSCENARIO to CommonRoad

Both OpenSCENARIO and CommonRoad offer freely available scenarios that can be easily customized and adapted as required. These scenarios are structured hierarchically but with different terminologies. This section concisely presents the conversion of logical scenarios from the OpenSCENARIO format to concrete CommonRoad scenarios. This is presented after a brief introduction of both formats, including an outline of their differences, followed by a detailed description of the conversion process.

A. OpenSCENARIO Format

The architecture of the OpenSCENARIO v1.2.0 format is presented in Fig. 1 as a unified modeling language (UML) class diagram. The header information for the scenario is contained in the module `FileHeader`. The class `ScenarioDefinition` groups the road network (class `RoadNetwork`), the configuration of obstacles (class `Entity`), and the container for the dynamic content (class `StoryBoard`). The `StoryBoard` is a core module of OpenSCENARIO as it specifies the temporal sequence of traffic situations and their triggers (class `Init`, `StartTrigger`, and `StopTrigger`) hierarchically into `Story`, `Act`, `Maneuver`, `Event`, and `Action`.

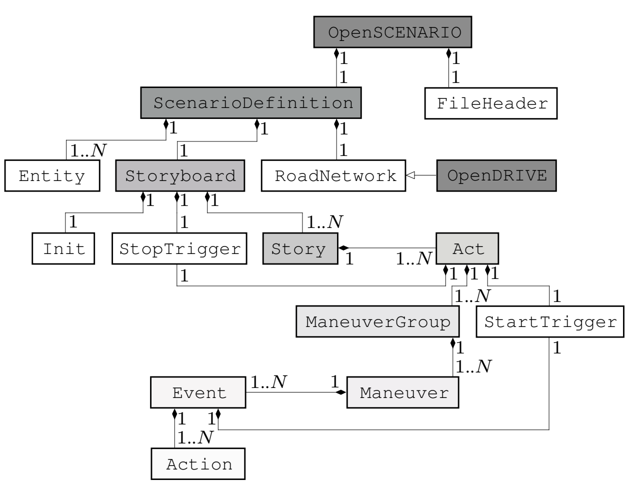
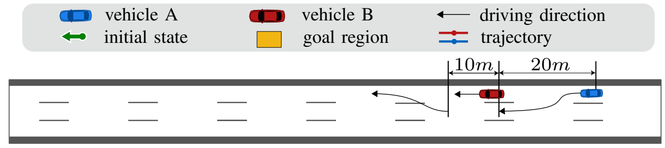
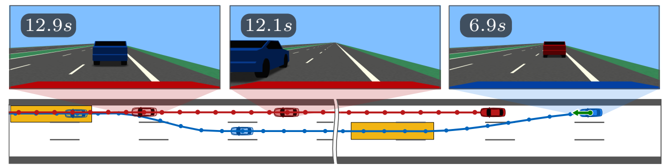


Fig. 1: UML class diagram of the OpenSCENARIO format. For brevity, we omit the nonessential classes such as those related to parameters.



(a) OpenSCENARIO.



(b) CommonRoad. The snapshots show the inside view of the vehicle, which is generated by esmini from the OpenSCENARIO file.

Fig. 2: Exemplary overtaking scenario.

Running example: In the scenario[11] in Fig. 2a, an overtaking `Story` is specified in the `Storyboard`. To achieve the overtaking task, two instances of the class `Event` are specified in a single `Maneuver` object: turn left and turn right. For the first event, vehicle A executes the `Action` of a lane change to the left when the relative longitudinal distance between the two vehicles falls below $20m$. For the second event, vehicle A is allowed to perform the lane change to the right when vehicle B is $10m$ ahead of vehicle A and the time elapsed since the last event is longer than $5s$.

B. CommonRoad Format

We present the UML class diagram of the CommonRoad v2023.2 format in Fig. 3. CommonRoad specifies a scenario (class `Scenario`) with a network of lanenets (class `LaneletNetwork`), one or several planning problems (class `PlanningProblem`), and obstacles (class `Obstacle`). Obstacles are characterized by their role, type (e.g., static or dynamic), shape, and initial state (class `State`). For dynamic obstacles (class `DynamicObstacle`), their movement over time is specified by trajectories (class `Trajectory`) that are a list of states, occupancy sets, or probability distributions. For motion planning, each *ego vehicle*, i.e., the vehicle to be controlled, has an initial state and one or several goal states (class `Goal`), which are described in the planning problem.

Running example: An overtaking scenario in CommonRoad format is shown in Fig. 2b. We can observe from the trajectories of both vehicles that vehicle A first changes its lane to the left of vehicle B and then returns to its initial lane. Assuming that vehicle A is the ego vehicle, one can model a

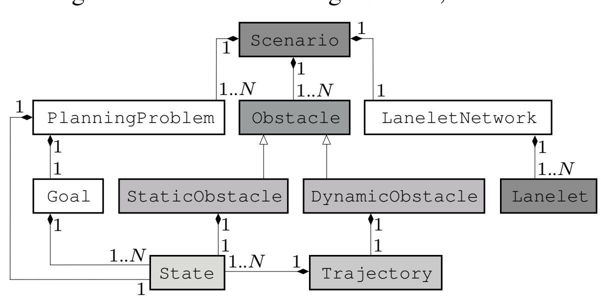


Fig. 3: UML class diagram of the CommonRoad format. Details of child elements are omitted for clarity.

[11] OpenSCENARIO ID: SimpleOvertake

Algorithm 1 OPENSCENARIOTOCOMMONROAD ▷ See Sec. II-D

Input: Scenario in the OpenSCENARIO format (denoted as OpenSCENARIO)
Output: Scenario in the CommonRoad format
1: lanelet_network: `LaneletNetwork` ← CONVERTOPENDRIVETOLANELETS(OpenSCENARIO.OpenDRIVE) ▷ See [44, Alg. 1]
2: obstacles: `List[Obstacle]` ← simulator.SIMULATE(OpenSCENARIO)
3: scenario: `Scenario` ← BUILDSCENARIO(obstacles, lanelet_network)
4: ego_vehicle: `Obstacle` ← FINDEGOVEHICLE(obstacles)
5: scenario.planning_problem: `PlanningProblem` ← BUILDPLANNINGPROBLEM(ego_vehicle) ▷ initial state and goal
6: **return** WRITETOXMLFILE(scenario)

`PlanningProblem` instance by combining its initial state and intermediate goals that are automatically or manually constructed.

C. OpenSCENARIO vs. CommonRoad

Both OpenSCENARIO and CommonRoad cover the design and implementation of a traffic scenario, i.e., what should happen and when it is executed. To simulate the `Storyboard` (cf. Fig. 1), OpenSCENARIO requires a director and a simulator core to govern the progress and execute control strategies based on the description. Thus, one cannot easily know how the scenario would look like unless simulating the traffic subject to OpenSCENARIO constraints. Moreover, different vehicle models, control strategies, and computer hardware can all contribute to varying traffic interactions within the simulation. In contrast, CommonRoad offers two ways to represent scenarios. The first option offers recordings of traffic situations, while the second option offers interactive simulations, i.e., other traffic participants react to the behavior of the ego vehicle through coupling with traffic simulators, such as SUMO [15]. On the other hand, OpenSCENARIO itself does not include driver models, vehicle dynamics, and cost functions as in CommonRoad, which currently limits its usage for many applications.

D. Implementation

The implementation of our OpenSCENARIO to CommonRoad converter is presented in Alg. 1. We begin by creating a lanelet network (see line 1) based on the OpenDRIVE file associated with OpenSCENARIO. This is accomplished by calling CONVERTOPENDRIVETOLANELETS described in [44, Alg. 1]. Afterwards, a simulation core is used, as described in detail later, to obtain trajectories of all simulated obstacles (see line 2). Then we build a CommonRoad scenario by aggregating the lanelet network and the obstacles (see line 3). To construct the planning problem, we either employ the predefined ego vehicle in OpenSCENARIO or allow the user to select a vehicle as the ego vehicle (see line 4). In line 5, the planning problem is formulated, e.g., based on the trajectory of the ego vehicle or the scenario descriptions. The conversion ends with writing the CommonRoad scenario in an XML file (see line 6).

To orchestrate and execute the dynamic elements defined by OpenSCENARIO (cf. Sec. II-C), we utilize esmini as the simulator in line 2 of Alg. 1 because:

1) reusing mature software reduces the complexity of the software structure and modularizes the framework;
2) esmini is more lightweight compared to, e.g., CARLA, yet has relatively high OpenSCENARIO coverage;
3) esmini has an interface for SUMO vehicle controllers and can send and receive OSI data, providing flexibility and real-time capabilities for traffic simulation; and
4) esmini provides a Python interface, which aligns with CommonRoad and many OpenSCENARIO tools.

During the simulation, the states of dynamic obstacles are collected at each frame in the global coordinate system of the converted lanelet network. By default, the esmini simulation ends as soon as all triggered elements (cf. Fig. 1) are completed. To prevent simulations from running indefinitely due to the absence of `StopTrigger` elements, we establish an upper time limit $t_{\max}$ for the scenario duration. Furthermore, we offer the possibility to increase the interactivity of the converter through the UDP interface of esmini, which facilitates the incorporation of external driver models.

After the simulation ends, the esmini Python binding is used to retrieve the information and states of all scenario

TABLE I: Conversion of obstacles from OpenSCENARIO to CommonRoad.

OpenSCENARIO	**CommonRoad**
Obstacle Type	
VEHICLE.CAR, VEHICLE.VAN	CAR
VEHICLE.TRUCK, VEHICLE.TRAILER, VEHICLE.SEMITRAILER	TRUCK
VEHICLE.BUS	BUS
VEHICLE.MOTORBIKE	MOTOCYCLE
VEHICLE.BICYCLE	BICYCLE
VEHICLE.TRAIN, VEHICLE.TRAM	TRAIN
PEDESTRIAN	PEDESTRIAN
MISC_OBJECT.BUILDING	BUILDING
MISC_OBJECT.TRAFFICISLAND	MEDIAN_STRIP
MISC_OBJECT.STREETLAMP	PILLAR
MISC_OBJECT.POLE, MISC_OBJECT.BARRIER, MISC_OBJECT.RAILING, MISC_OBJECT.SOUNDBARRIER	ROAD_BOUNDARY
MISC_OBJECT.PATCH	CONSTRUCTION_ZONE
others	UNKNOWN
State	
state.timestamp	state.time_step
[state.x, state.y]	state.position
state.h	state.orientation
state.speed	state.velocity
state.wheelAngle	state.steering_angle
state.h_rate	state.yaw_rate
Shape	
state.length	obstacle_shape.length
state.width	obstacle_shape.width

objects and convert them to CommonRoad types. Tab. I lists the transformation relation from OpenSCENARIO to CommonRoad obstacles. To match the required time step size Δt of the CommonRoad scenario, esmini trajectories may need to be resampled, e.g., using methods from [16, Sec. 3.3].

III. Numerical Experiments

To demonstrate the usefulness of our converter, we convert 54 openly accessible OpenSCENARIO scenarios from:

I. OpenSCENARIO standard examples,
II. the esmini demonstration package, and
III. automated lane keeping scenarios[12].

We exclude OpenSCENARIO files that contain only parameter values and allow certain elements to be reusable. All conversions are performed on a computer with an Intel Core i7-1165G7 CPU and 16 GB of memory. The parameters for the converter are listed in Tab. II.

A. Conversion Statistics

The conversion statistics are listed in Tab. II. Our converter successfully transformed all considered OpenSCENARIO files with an average conversion time of $31.74s$. We observed that the duration of the simulation closely correlates with the complexity of the scenario and the map size. As a result, our converter proves to be an effective tool for converting the OpenSCENARIO description into the CommonRoad format.

B. Scenario Evaluation on CommonRoad

We demonstrate the practicality of our converter by evaluating an OpenSCENARIO scenario[13] using CommonRoad tools. In this scenario, the ego vehicle and a pedestrian follow predefined routes, where the pedestrian jaywalks. Fully comprehending the scenario based solely on the OpenSCENARIO file is challenging. To obtain a deeper understanding, we use our converter to simulate the traffic and show the simulated result in Fig. 4a. Afterwards, we evaluate the scenario with CommonRoad tools in the following paragraphs:

a) Collision Checking: we use the open-source toolbox CommonRoad Drivability Checker [17] to check the drivability of the trajectory of the ego vehicle. It can be verified that this trajectory is kinematically feasible; however, it collides with the pedestrian at $5.5s$. The collision occupancies are highlighted in Fig. 4b.

TABLE II: Settings and conversion statistics for the conversion.

Parameter	Value
esmini Δt	$0.01s$
CommonRoad Δt	$0.1s$
$t_{\max}$	$60s$

Source	Success Rate	Avg. Conversion Time	Avg. Scenario Duration
I	100.0% (13/13)	$10.68s$	$53.61s$
II	100.0% (26/26)	$18.20s$	$30.96s$
III	100.0% (15/15)	$66.34s$	$40.64s$

[12] https://github.com/asam-oss/OSC-ALKS-scenarios
[13] OpenSCENARIO ID: pedestrian_collision

b) Motion Planning: using CommonRoad, motion planners can be easily benchmarked. As shown in Fig. 4c, the popular motion planner described in [46] successfully avoids collision with the pedestrian by maneuvering the ego vehicle to the left.

c) Criticality Comparison: CommonRoad is also equipped with various criticality measures through its tool CommonRoad-CriMe [20], which are designed to objectively evaluate the safety and threat level of traffic scenarios. As an example, we use time-to-collision (TTC), worst-time-to-collision (WTTC), time-to-react (TTR), drivable area (DA), brake threat number (BTN), and steer threat number (STN) to evaluate the converted scenario. The curves in Fig. 4d all show that the scenario is getting more critical over time. The TTR curve indicates that there are no available evasive maneuvers to avoid the collision after $4.8s$.

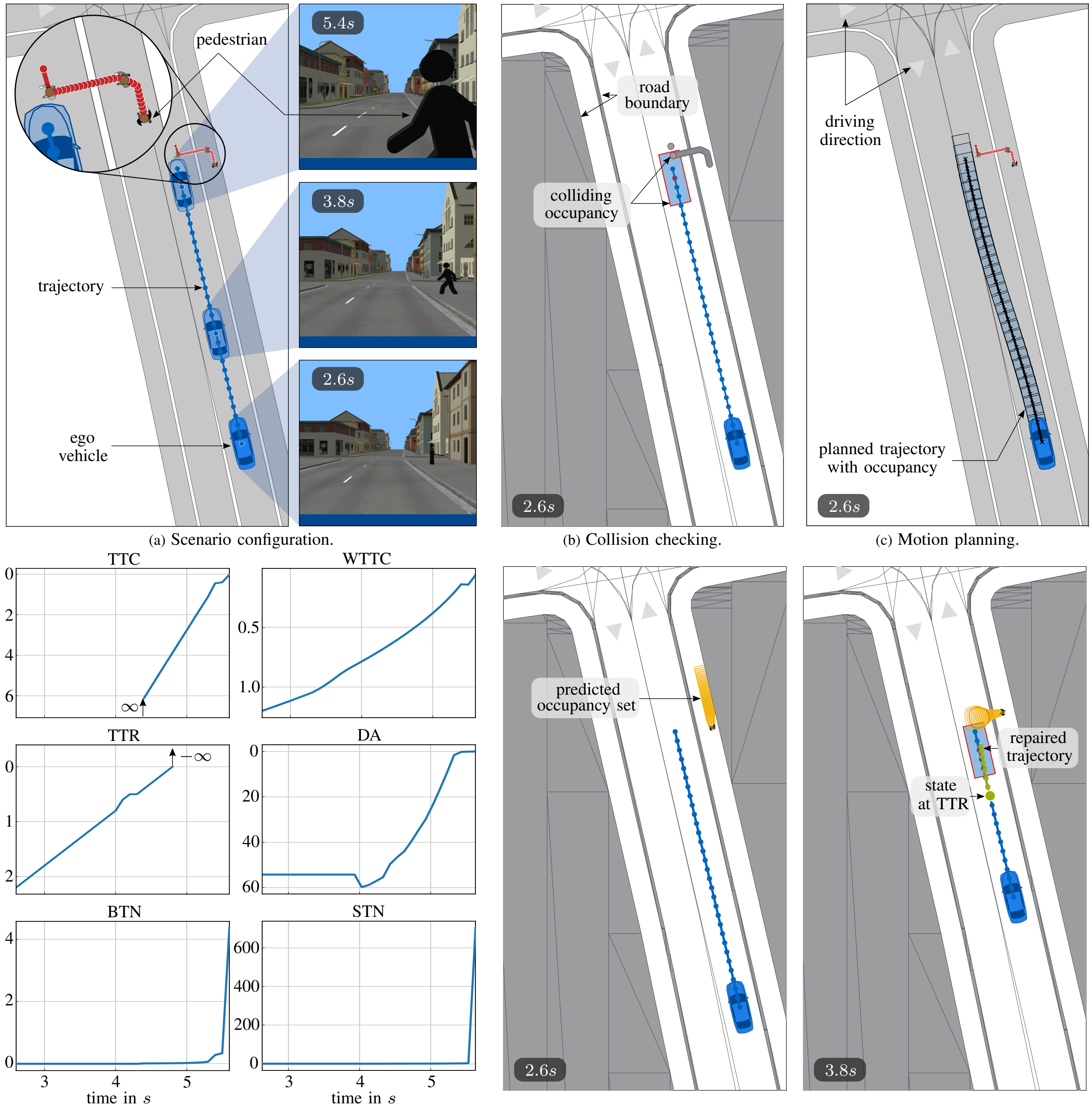
d) Safety Verification and Trajectory Repairing: CommonRoad offers the tool SPOT [18] to predict the occupancy set of obstacles based on legal behaviors. With SPOT, we can safeguard the ego vehicle within a given time interval by ensuring that the planned trajectory is collision-free against the predicted occupancy set [21]. As a result, in Fig. 4e, the ego vehicle is considered legally safe at $2.6s$ along the intended trajectory but not at $3.8s$, as the latter could pose a danger to the pedestrian when it is inattentive and jaywalks. To efficiently ensure safety for the situation at $3.8s$, we employ the trajectory repairing method described in [47] and [48]. This approach keeps the trajectory before the TTR unchanged while repairing the remaining part. Thereby the ego vehicle fully brakes to prevent any potential harm to the pedestrian.

IV. Conclusions

This paper introduces the first publicly available converter from OpenSCENARIO to CommonRoad. By triggering the dynamic elements defined in OpenSCENARIO, the logical scenarios are concretized into CommonRoad format, incorporating predefined interactions between vehicles. We aim to foster the development, testing, and validation of autonomous driving systems by providing an open-source solution for converting scenarios between different formats. This also serves to bridge the gap between academia and industry, thereby promoting the advancement of technology in the field of autonomous driving.

Acknowledgments

We would like to express our sincere appreciation to Emil Knabe for his invaluable contribution in reviewing and accepting the proposed changes to the esmini interface, and to Sebastian Maierhofer for maintaining the converter from OpenDRIVE to lanelets. Furthermore, the authors gratefully acknowledge partial financial support by the German Federal Ministry for Digital and Transport (BMDV) within the project *Cooperative Autonomous Driving with Safety Guarantees* (KoSi).



(a) Scenario configuration. (b) Collision checking. (c) Motion planning.

(d) Criticality measures. (e) Safety verification and trajectory repairing.

Fig. 4: Evaluation results with CommonRoad tools. We only display the scenario information between $2.6s$ and $5.6s$. (a) shows the configuration of the converted CommonRoad scenario, with snapshots captured from the inside view of the ego vehicle during esmini simulation at three time steps. Collision checking and motion planning results are presented in (b) and (c), respectively. To provide clear insights, the criticality of the scenario is plotted on the vertical axis of the graph, with an upward trend indicating increasing criticality, as shown in (d). Finally, (e) displays the safety verification results at both $2.6s$ and $3.8s$, where the trajectory is repaired if the intended trajectory is not legally safe.

References

[1] T. Menzel, G. Bagschik, and M. Maurer, "Scenarios for development, test and validation of automated vehicles," in *Proc. the IEEE Intell. Veh. Symp.*, 2018, pp. 1821–1827.

[2] M. Althoff, M. Koschi, and S. Manzinger, "CommonRoad: Composable benchmarks for motion planning on roads," in *Proc. the IEEE Intell. Veh. Symp.*, 2017, pp. 719–726.

[3] S. Riedmaier, T. Ponn, D. Ludwig, B. Schick, and F. Diermeyer, "Survey on scenario-based safety assessment of automated vehicles," *IEEE access*, vol. 8, pp. 87 456–87 477, 2020.

[4] R. Krajewski, J. Bock, L. Kloeker, and L. Eckstein, "The highD dataset: A drone dataset of naturalistic vehicle trajectories on German highways for validation of highly automated driving systems," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2018, pp. 2118–2125.

[5] W. Zhan, L. Sun, D. Wang, H. Shi, A. Clausse, M. Naumann, J. Kümmerle, H. Königshof, C. Stiller, A. de La Fortelle, and M. Tomizuka, "INTERACTION dataset: An INTERnational, Adversarial and Cooperative moTION dataset in interactive driving scenarios with semantic maps," *arXiv preprint arXiv:1910.03088*, 2019.

[6] J. Bock, R. Krajewski, T. Moers, S. Runde, L. Vater, and L. Eckstein, "The inD dataset: A drone dataset of naturalistic road user trajectories

at German intersections," in *Proc. the IEEE Intell. Veh. Symp.*, 2020, pp. 1929–1934.
[7] R. Krajewski, T. Moers, J. Bock, L. Vater, and L. Eckstein, "The rounD dataset: A drone dataset of road user trajectories at roundabouts in Germany," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2020, pp. 1–6.
[8] T. Moers, L. Vater, R. Krajewski, J. Bock, A. Zlocki, and L. Eckstein, "The exiD dataset: A real-world trajectory dataset of highly interactive highway scenarios in Germany," in *Proc. the IEEE Intell. Veh. Symp.*, 2022, pp. 958–964.
[9] Y. Xu, W. Shao, J. Li, K. Yang, W. Wang, H. Huang, C. Lv, and H. Wang, "SIND: A drone dataset at signalized intersection in China," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2022, pp. 2471–2478.
[10] H. Caesar, J. Kabzan, K. S. Tan, W. K. Fong, E. Wolff, A. Lang, L. Fletcher, O. Beijbom, and S. Omari, "NuPlan: A closed-loop ML-based planning benchmark for autonomous vehicles," in *CVPR Workshop on Autonomous Driving: Perception, Pred. and Plan.*, 2021.
[11] L. Gressenbuch, K. Esterle, T. Kessler, and M. Althoff, "MONA: The munich motion dataset of natural driving," in *Proc. the IEEE Int. Conf. on Intell. Transp. Sys.*, 2022, pp. 2093–2100.
[12] S. Maierhofer, M. Klischat, and M. Althoff, "Commonroad Scenario Designer: An open-source toolbox for map conversion and scenario creation for autonomous vehicles," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2021, pp. 3176–3182.
[13] M. Klischat and M. Althoff, "Generating critical test scenarios for automated vehicles with evolutionary algorithms," in *Proc. the IEEE Intell. Veh. Symp.*, 2019, pp. 2352 – 2358.
[14] M. Klischat, E. I. Liu, F. Holtke, and M. Althoff, "Scenario factory: Creating safety-critical traffic scenarios for automated vehicles," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2020, pp. 1–7.
[15] M. Klischat, O. Dragoi, M. Eissa, and M. Althoff, "Coupling SUMO with a motion planning framework for automated vehicles," in *SUMO User Conf.*, 2019, pp. 1–9.
[16] X. Wang, A.-K. Rettinger, M. T. B. Waez, and M. Althoff, "Coupling Apollo with the CommonRoad motion planning framework," in *Proc. the FISITA Web Congress*, 2020, p. 24.
[17] C. Pek, V. Rusinov, S. Manzinger, M. C. Üste, and M. Althoff, "CommonRoad Drivability Checker: Simplifying the development and validation of motion planning algorithms," in *Proc. the IEEE Intell. Veh. Symp.*, 2020, pp. 1013–1020.
[18] M. Koschi and M. Althoff, "SPOT: A tool for set-based prediction of traffic participants," in *Proc. the IEEE Intell. Veh. Symp.*, 2017, pp. 1686–1693.
[19] E. I. Liu, G. Würsching, M. Klischat, and M. Althoff, "CommonRoad-Reach: A toolbox for reachability analysis of automated vehicles," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2022, pp. 2313–2320.
[20] Y. Lin and M. Althoff, "CommonRoad-CriMe: A toolbox for criticality measures of autonomous vehicles," in *Proc. the IEEE Intell. Veh. Symp.*, 2023.
[21] C. Pek, S. Manzinger, M. Koschi, and M. Althoff, "Using online verification to prevent autonomous vehicles from causing accidents," *Nature Machine Intell.*, vol. 2, no. 9, pp. 518–528, 2020.
[22] T. Nyberg, C. Pek, L. Dal Col, C. Norén, and J. Tumova, "Risk-aware motion planning for autonomous vehicles with safety specifications," in *Proc. the IEEE Intell. Veh. Symp.*, 2021, pp. 1016–1023.
[23] J. Li, X. Xie, Q. Lin, J. He, and J. M. Dolan, "Motion planning by search in derivative space and convex optimization with enlarged solution space," in *Proc. the IEEE Int. Conf. on Intell. Robots and Sys.*, 2022, pp. 13 500–13 507.
[24] M. Geisslinger, F. Poszler, and M. Lienkamp, "An ethical trajectory planning algorithm for autonomous vehicles," *Nature Machine Intell.*, vol. 5, no. 2, pp. 137–144, 2023.
[25] A. Zanardi, G. Zardini, S. Srinivasan, S. Bolognani, A. Censi, F. Dörfler, and E. Frazzoli, "Posetal games: Efficiency, existence, and refinement of equilibria in games with prioritized metrics," *IEEE Robotics and Automation Letters*, vol. 7, no. 2, pp. 1292–1299, 2021.
[26] L. Xiong, L. Cao, B. Leng, and M. Liu, "Noncooperative-game-based intelligent vehicle decision method for lane-changing interactive behavior," in *Proc. the IEEE Chinese Control and Decision Conf.*, 2022, pp. 28–34.
[27] A. Zanardi, S. Bolognani, A. Censi, F. Dorfler, and E. Frazzoli, "Factorization of dynamic games over spatio-temporal resources," in *Proc. the IEEE Int. Conf. on Intell. Robots and Sys.*, 2022, pp. 13 159–13 166.
[28] X. Wang, H. Krasowski, and M. Althoff, "Commonroad-RL: A configurable reinforcement learning environment for motion planning of autonomous vehicles," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2021, pp. 466–472.
[29] S. Khaitan and J. M. Dolan, "State dropout-based curriculum reinforcement learning for self-driving at unsignalized intersections," in *Proc. the IEEE Int. Conf. on Intell. Robots and Sys.*, 2022, pp. 12 219–12 224.
[30] E. Meyer, M. Brenner, B. Zhang, M. Schickert, B. Musani, and M. Althoff, "Geometric deep learning for autonomous driving: Unlocking the power of graph neural networks with CommonRoad-Geometric," in *Proc. the IEEE Intell. Veh. Symp.*, 2023.
[31] J. Dobberstein, J. Bakker, L. Wang, T. Vogt, M. Düring, L. Stark, J. Gainey, A. Prahl, R. Mueller, and G. Blondelle, "The Eclipse working group openPASS–an open source approach to safety impact assessment via simulation," in *Proc. the Int. Technical Conf. on the Enhanced Safety of Veh.*, 2017.
[32] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, "CARLA: An open urban driving simulator," in *Conf. on robot learning*, 2017, pp. 1–16.
[33] A. Tenbrock, A. König, T. Keutgens, and H. Weber, "The ConScenD dataset: Concrete scenarios from the highD dataset according to ALKS regulation UNECE R157 in OpenX," in *Proc. the IEEE Intell. Veh. Symp. Workshops*, 2021, pp. 174–181.
[34] D. Karunakaran, J. S. Berrio, S. Worrall, and E. Nebot, "Parameterisation of lane-change scenarios from real-world data," in *Proc. the IEEE Int. Conf. on Intell. Transp. Syst.*, 2022, pp. 2607–2613.
[35] H. Chen, H. Ren, R. Li, G. Yang, and S. Ma, "Generating autonomous driving test scenarios based on OpenSCENARIO," in *Proc. the IEEE Int. Conf. on Dependable Sys. and Their App.*, 2022, pp. 650–658.
[36] D. Salles, L. Lang, M. Kehrer, and H.-C. Reuss, "A modular co-simulation framework with open source software and automotive standards," in *Int. Stuttgarter Symp.: Automobil-und Motorentechnik*. Springer, 2022, pp. 207–223.
[37] R. Queiroz, T. Berger, and K. Czarnecki, "GeoScenario: An open DSL for autonomous driving scenario representation," in *Proc. the IEEE Intell. Veh. Symp.*, 2019, pp. 287–294.
[38] D. J. Fremont, T. Dreossi, S. Ghosh, X. Yue, A. L. Sangiovanni-Vincentelli, and S. A. Seshia, "Scenic: A language for scenario specification and scene generation," in *Proc. the ACM SIGPLAN Conf. on Prog. Language Design and Implementation*, 2019, pp. 63–78.
[39] B. Schütt, T. Braun, S. Otten, and E. Sax, "SceML: A graphical modeling framework for scenario-based testing of autonomous vehicles," in *Proc. the ACM/IEEE Int. Conf. on Model Driven Engineering Languages and Sys.*, 2020, pp. 114–120.
[40] X. Zhang, S. Khastgir, and P. Jennings, "Scenario description language for automated driving systems: A two level abstraction approach," in *Proc. the IEEE Int. Conf. on Sys., Man, and Cybernetics*, 2020, pp. 973–980.
[41] D. Du, J. Chen, M. Zhang, and M. Ma, "Towards verified safety-critical autonomous driving scenario with ADSML," in *Proc. the IEEE Annual Computers, Software, and App. Conf.*, 2021, pp. 1333–1338.
[42] R. Majumdar, A. Mathur, M. Pirron, L. Stegner, and D. Zufferey, "Paracosm: A test framework for autonomous driving simulations," in *Proc. the Fundamental Approaches to Software Engineering*. Springer, 2021, pp. 172–195.
[43] C. Chang, D. Cao, L. Chen, K. Su, K. Su, Y. Su, F.-Y. Wang, J. Wang, P. Wang, J. Wei, *et al.*, "MetaScenario: A framework for driving scenario data description, storage and indexing," *IEEE Trans. on Intell. Veh.*, vol. 8, no. 2, pp. 1156–1175, 2023.
[44] M. Althoff, S. Urban, and M. Koschi, "Automatic conversion of road networks from OpenDRIVE to lanelets," in *Proc. the IEEE Int. Conf. on Service Operations and Logistics, and Info.*, 2018, pp. 157–162.
[45] P. Bender, J. Ziegler, and C. Stiller, "Lanelets: Efficient map representation for autonomous driving," in *Proc. the IEEE Intell. Veh. Symp.*, 2014, pp. 420–425.
[46] M. Werling, S. Kammel, J. Ziegler, and L. Gröll, "Optimal trajectories for time-critical street scenarios using discretized terminal manifolds," *The Int. J. of Robotics Research*, vol. 31, no. 3, pp. 346–359, 2012.
[47] Y. Lin, S. Maierhofer, and M. Althoff, "Sampling-based trajectory repairing for autonomous vehicles," in *Proc. the IEEE Int. Conf. on Intell. Transp. Sys.*, 2021, pp. 572–579.
[48] Y. Lin and M. Althoff, "Rule-compliant trajectory repairing using satisfiability modulo theories," in *Proc. the IEEE Intell. Veh. Symp.*, 2022, pp. 449–456.