2020 IEEE Intelligent Vehicles Symposium (IV)
October 20-23, 2020. Las Vegas, USA

CSG: Critical Scenario Generation from Real Traffic Accidents

Zhang Xinxin[1], Li Fei[2], Wu Xiangbin[3]

Abstract— **Autonomous driving (AD) is getting closer to our life, but the severe traffic accidents of autonomous vehicle (AV) happened in the past several years warn us that the safety of AVs is still a big challenge for the AD industry. Before volume production, the automotive industry and regulators must ensure the AV can deal with dangerous scenarios. Although road test is the most common method to test the performance and safety of an AV, it has some manifest disadvantages, e.g., highly risky and unrepeatable, low efficiency and lack of useful critical scenarios. Critical-scenario-based simulation can effectively address these problems and become an important complement to road test. In this paper, we present a novel approach to extract critical scenarios from real traffic accident videos and re-generate them in a simulator. We also introduce our integrated toolkit for scenario extraction and scenario test. With the toolkit, we can build a critical scenario library quickly and use it as a benchmark for AV safety assessment, among other purposes. On top of this, we further introduce our safety assessment criteria and scoring method.**

I. INTRODUCTION

More than 100 years after the automobile was invented, the traditional automotive industry is being dramatically reshaped by the revolutionary innovation of Autonomous Driving, or AD. Some leading automotive companies have developed their autonomous driving system (ADS) and test AVs on the road for millions of miles. It seems that the new era of AD will be coming soon. But in the past several years, some severe traffic accidents of AV raised people's serious concern for the safety of AV. In these accidents, the drivers or pedestrians were seriously injured and even unfortunately died. These accidents reveal a big challenge for AD, that is how to evaluate the AV's safety before its volume production to ensure they can drive on the road safely?

For L1 and L2 level autonomous system, the vehicle is still charged by a human driver assisted by an Advanced Driver Assistance System (ADAS). The ADAS functionalities, including Adaptive Cruise Control (ACC), Lane Keeping Assist (LKA), Forward Collision Warning (FCW), and Automatic Emergency Braking (AEB), etc., can help to dramatically improve the safety in some dangerous situations. These ADAS functionalities can be individually tested and the test condition is easy to construct. But from level 3 to level 5, all driving functions will be taken over by the ADS. ADS is a complex computer system equipped with many kinds of sensors, e.g., camera, lidar, radar, GPS, and IMU. ADS processes the mass data generated by sensors to monitor and perceive the driving environment, localize itself and plan its trajectory and actions. Compared to ADAS, ADS is not a simple combination of many small functionalities, but an undivided system, which can only be evaluated under complex driving environments rather than a single test condition.

But the complex driving environments are hard to construct and are impossible to enumerate. Even though some AD companies, e.g., Waymo [26], Uber, and Tesla, have run their self-driving cars on the road for millions of miles to evaluate the performance and safety, it is far from the necessary mileage to prove its safety [25]. The road test is very important and essential, but it has its own disadvantages. The road test is highly risky and unrepeatable, and the test results cannot be compared with others because of the different road conditions. More importantly, the road test is low efficient, because most test miles are just simple repeats, and lack of unknown critical scenarios.

To address these problems, the AD industry and academics want to find other effective methods as complements to the road test. Simulation is a possible solution because it is safe, efficient and low cost. In fact, the software driving simulators have been widely used by AD companies, e.g. Waymo's CarCraft [26], Baidu's Apollo [30], Nvidia's Drive Constellation [1], Microsoft's AirSim [2] and Intel's CARLA [3], etc. With a popular 3D graphic engine and powerful GPU configuration, these simulators can emulate very realistic street views and road conditions. AV developers and researchers can run their virtual cars in the virtual environment for millions of miles a day. But compared with the 3D static scene rendering, the more difficult thing is to emulate the critical scenarios, which can be used to examine the ADS's emergency response to hazard and find out the hidden problems which will potentially lead to a crash.

In this paper, we propose a critical-scenario-based methodology as a complement to road test, and present a novel approach, Critical Scenario Generation (CSG), to generate the critical scenarios in driving simulators for safety tests. Different from previous work, our approach uses a set of computer vision algorithms to extract the critical scenario directly from a real traffic accident video and then regenerates it in a simulator. With the toolkit we developed, we can build a critical scenario library and use it as a benchmark to assess the safety of AV. In this paper, we also discuss the key design choices, safety assessment criteria, and potential applications.

At the current stage of our work, we are focusing on applying critical scenarios to test the planning part of an

[1] Research scientist of Intelligent Driving Lab, Intel Labs China, Beijing 100190, China xinxin.zhang@intel.com
[2] Research scientist of Intelligent Driving Lab, Intel Labs China, Beijing 100190, China fei.a.li@intel.com
[3] Research director of Intelligent Driving Lab, Intel Labs China, Beijing 100190, China xiangbin.wu@intel.com

ADS, which typically includes functions such as prediction, routing, decision making and motion planning, as it is more likely to fail in a sudden and dangerous situation. The testing of the perception part is beyond the scope of this paper, as there are many available datasets for that.

II. CURRENT SCENARIO GENERATION METHODS

In this paper, a critical scenario refers to a certain dangerous situation in which the ego vehicle may make an unsafe decision that is going to lead to an accident, or in which the ego vehicle can be involved and must take appropriate countermeasures to avoid an imminent collision or other loss. In the foreseeable future, the AVs will coexist with the human-driven vehicles on roads for a long time. But the behaviors of human-driven vehicles and pedestrians have considerable uncertainties typically because of reckless driving, violating traffic regulations intentionally, negligence, loss of vehicle control, absent-mindedness or some other reasons. These uncertainties will result in many unpredictable dangerous situations for AV. The difficulty of critical scenario generation is to reproduce these uncertainties, complexities, and diversities altogether.

A first method of generating a critical scenario is to orchestrate it manually. The open-source project Scenario Runner [5] orchestrates a scenario by organizing a sequence of trigger conditions and corresponding atomic behaviors in a behavior tree. GeoScenario [4] proposed a Domain-Specific Language for scenario representation and developed a tool to design the scenario on the real-world map. MATLAB also provides a tool named Driving Scenario Designer [6], to create and edit a scenario by defining the actors, road network and waypoints, etc. However, the manual method is time-consuming and has obvious artifacts. More importantly, the manually designed scenarios lack the diversity and complexity of real accident scenarios. Many dangerous situations are beyond our imagination and hard to design by mere expertise.

Another method is to collect critical scenario data through naturalistic driving studies (NDS). NDS depends on vehicles equipped with an on-board data acquisition system to collect high-accuracy vehicle kinematics during daily driving [7]. The detailed information contained in NDS data, including road environment, vehicle movement and driver behavior can be used to rebuild the safety-critical scenario. AADS [8] collected 1000km of trajectories for all kinds of moving traffic agents under a variety of lighting conditions and traffic densities, and then used the dataset to generate new plausible traffic flow in the simulator. [16] uses some trigger criteria like high lateral and longitudinal acceleration and low time-to-collision (TTC), to identify and extract the safety-critical events from the Shanghai Naturalistic Driving Study (SH-NDS) [9], and then rebuild them in a simulator. This method is time-consuming and costly. For SH-NDS, fifty-seven drivers participated in the study and drove 161,055 kilometers in total during the three-year study.

Different from these previous methods, CSG is aiming to extract critical scenarios from real traffic accident videos, and finally build a large-scale critical scenario library as a benchmark for AV safety assessment. Our approach has these advantages:

1) Compared with mileage-based methodology, the critical-scenario-based methodology can greatly reduce the cost and time required by the development and test of a new ADS system, and it is an important complement to road test. Especially, CSG is highly efficient, repeatable and comparable based on the same test conditions and uniform evaluation criteria.
2) Compared with synthetic scenario generation, CSG approach can restore the complexity and diversity of real-world scenarios and minimize artifacts. We extract raw information from the accident, including actors, road networks and vehicle kinematics, among others. The virtual scenarios have very similar characteristics as real accidents.
3) Compared with NDS method, it is easier to build the critical scenario library using CSG approach. The existing accident videos can be used to build a large dataset, and more importantly, its scale can easily grow because CSG approach uses crowd-sourcing to collect accident videos without much dependency on specialized data acquisition system. In recent years the computer vision (CV) technology has made great progress, and deep neural network (DNN) can be used for object detection, classification, segmentation effectively [10]. With novel CV and DNN algorithms, CSG approach can be highly efficient and automatic.

III. SYSTEM ARCHITECTURE AND KEY DESIGN CHOICES

In this section, we will introduce the system architecture of CSG and the key design choices.

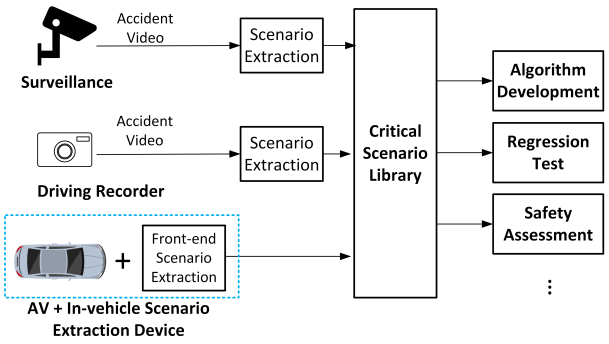

Fig. 1. System architecture

As shown in Figure 1, CSG has three main design goals:

1) Critical scenario extraction method and toolkit. The toolkit should extract critical scenario information from real accident videos accurately as much as possible. Each scenario should be described in a standard format to achieve interoperability.

2) Critical scenario library. With the scenario extraction tool, a large-scale critical scenario library can be built as the baseline to benchmark and evaluate the safety of AV. The library should be independent of simulators to maximize versatility.
3) Safety assessment criteria. Based on the library, novel safety criteria and the scoring method can be established, and be used for the safety assessment of a new car, regression test and new algorithm development, etc.

A. Data Source

Currently, most accident videos are recorded by roadside surveillance cameras or driving recorders mounted in the car, and the vehicles involved are almost all human-driven vehicles. But with the deployment of AVs, e.g. Robo-Taxi, we can collect the AV involved accidents which are equally valuable for ADS development and test.

AV is usually equipped with many sensors including cameras, lidars, GPS, IMU etc., and thus can record accurately more details of the accident, not just the video. AVs can analyze and process the accident data locally, especially, remove the privacy data contained in the original video, and then upload the extracted scenario to the cloud automatically. A similar principle has been used by Mobileye in its Road-Experience Management (REM) crowd-sourcing HD-Map solution.

B. Scenario Specification

A complete scenario contains three kinds of critical elements: road network, environment, and traffic dynamics, as shown in Figure 2.

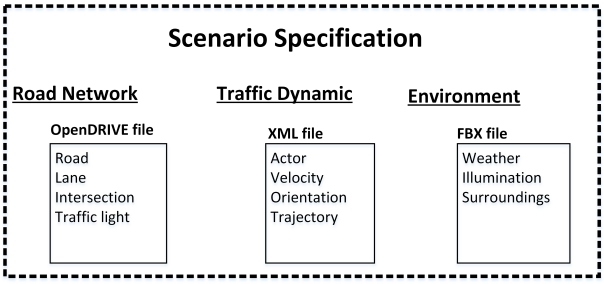

Fig. 2. Scenario specification

Road network describes the topology information about the road, and their semantics, e.g., lanes, road markings, road connections, traffic lights and other details [11]. For most traffic accident videos, it is next to impossible to know where the accident happened, so we could not get the road network from the existing real-world map, like what GeoScenario [4] does. The road network must be extracted from the video directly. In CSG we use OpenDRIVE [18] format to save the road network information because it is open and widely supported by many simulators.

Traffic dynamics includes the description of all traffic participants (cars, pedestrians, motorcycles, etc.), and their movement information, such as trajectory, velocity, orientation, etc. Menzel et al propose three abstraction levels for scenarios: functional, logical and concrete scenarios [12]. Obviously, the traffic dynamics we extract belong to the concrete level. For traffic dynamics, we basically leverage the XML-based description format used by CARLA scenario engine [5], but with some extensions.

The environment information includes weather conditions, illumination, surrounded buildings and trees, etc. These elements are very useful for evaluating perception modules. Usually, the environment information is involved in a 3D model file, e.g. FBX [28], which is a semi-open standard. Currently, it is very difficult to automatically extract the 3D model of surrounding buildings, trees and other objects. Thus, we use third-party software for 3D modeling and scenery rendering, and then export the FBX file. The FBX file will be imported into a simulator to create the 3D scenery.

C. Scenario Enhancement

In each traffic accident, the movements of all actors jointly form a dangerous situation and finally led to a collision. In general, among these actors the car driving normally in the lane will be assigned as ego vehicle, the car or walker colliding with the ego vehicle will be assigned as the troublemaker, and the others will play as background traffic flow. But in a simulation test, the ego vehicle will be taken over by the ADS, which brings a potential problem: the scenario becomes no longer dangerous to the virtual ego vehicle if it has totally different dynamics from the physical ego vehicle in the real accident.

To solve this problem, [4] and [5] use a triggers & actions scheme in the virtual scenario, and the scenario is executed only when the trigger is activated, e.g. the ego vehicle reaches a pre-defined speed or location. But the problem with this approach is that it is very difficult to determine proper and accurate trigger conditions due to the complexity of the real accident scenarios. In our study, we adopt a different approach to solve the aforementioned problem, which is to impose some limitations to the virtual ego vehicle's dynamics, including its initial speed, target speed and waypoints, and meanwhile, we adopt the parameter variation method to derive lots of test cases from the original critical scenario. With these test cases, we can discover the safety boundary of the ADS under test and make it more robust.

D. Scenario Library Management

National Highway Traffic Safety Administration (NHTSA) defines a widely cited pre-crash scenario typology which statistically describes 37 pre-crash scenarios for light vehicles [13]. Currently we also use this typology to manage our scenario library. But in fact, each country has its own unique traffic system and road conditions, thus the scenario typology can also be different. For example, in China many traffic accidents are caused by reckless motorcycles and pedestrians, which is very different from that in America. In the future we will use different scenario typologies for different country.

Besides that, all scenarios can be classified into clusters by certain clustering algorithms, for example, K-means clustering, mixed Gaussian model and hierarchical clustering. This method can help us to evaluate the similarity of accidents and identify the most important factors that contribute to the accidents.

E. Scenario Engine

Most simulators only provide a virtual driving environment without scenario test functionality [3] or can only execute very limited preset scenarios [30]. To address the problem, we need to implement an external scenario engine to parse the scenario specification and then inject it into a simulator for execution. The scenario engine is tightly coupled with the simulator because each simulator has its own unique features and user interface.

F. Scoring and Safety Assessment

There are some widely used criteria for scoring. Tong et al. [14] propose a key performance indicator (KPI) which includes safety, comfort, natural driving and ecology. Currently in our study, we pay more attention to the safety and efficiency of AV.

The most notable safety indicator is whether a collision happens. Besides that, some surrogate safety measurement (SSM), such as the deceleration rate to avoid a crash, Time to Collision (TTC) and Time Integrated TTC (TIT), can be used to evaluate the potential crash risk. Compliance with traffic rules can also be adopted as a safety indicator. Efficiency indicator means the AV should not guarantee safety at the cost of significantly reducing traffic efficiency, e.g., driving through the test scenario at an unreasonable low speed. The efficiency indicator can also be used to prevent possible cheating in the critical scenario test.

The weighted sum of all the above indicators can be used for scoring for a single scenario. But the safety assessment should be determined by the synthesis score of all scenarios in the library, rather than a single scenario. In our current study, the synthesis score is defined as:

$$S_{syn} = \sum_{c=1}^{M} (W_c \frac{1}{N_c} \sum_{i=1}^{N_c} S_{c,i}) \quad (1)$$

Where S_{syn} is the synthesis score; M is the number of all categories in the scenario typology; W_c is the weight of category c; N_c is the number of all scenarios in category c; $S_{c,i}$ is the score of scenario i in category c.

IV. CRITICAL SCENARIO GENERATION TOOLKIT

We developed an integrated toolkit to achieve the end-to-end processing flow, from scenario extraction, scenario library management to scenario test, as shown in Figure 3. It should be noted that under the current framework each function module in the toolkit is reusable and replaceable, and thus easy for further optimization. In the following sections we will introduce in detail of each function module in the toolkit.

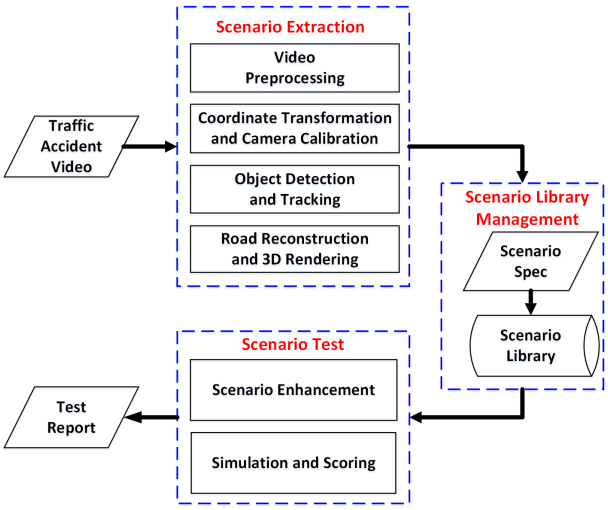

Fig. 3. Processing pipeline of CSG toolkit. The extracted scenario elements are manually checked and corrected if needed at each step.

A. Video Preprocessing

By now the accident videos we collected are all recorded in China where the road conditions are very challenging for AV. Among them, those videos of poor quality or improper perspective will be manually excluded first. Technically the surveillance video is more convenient to process than driving recorder video because it has a fixed perspective and it is easier to extract the road networks and vehicle kinematics. Currently, we just focus on the surveillance videos.

Our toolkit implements a Multi-Scale Retinex algorithm for image enhancement, which is essential when the video is recorded in weak illumination or poor resolution.

B. Coordinate Transformation and Camera Calibration

Coordinate transformation is a fundamental step in the whole pipeline, and it maps each pixel in the 2D video image (image coordinate system) to a point in the 3D space (world coordinate system) through a camera matrix, as shown in Figure 4.

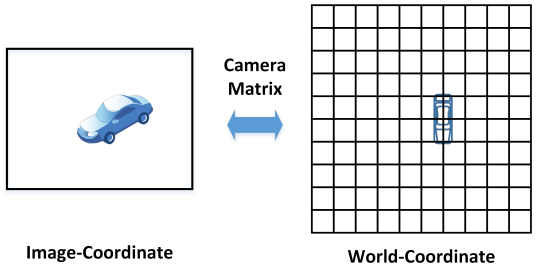

Fig. 4. Coordinate transformation

The scene information, including road network, actor's location, trajectory, velocity, etc., are all calculated in the world coordinate. We can easily calculate the camera matrix if we know the extrinsic and intrinsic parameters of the camera [15], such as the focal length, the skew coefficient, the rotation and translation, etc. But unfortunately, many of

the video clips we access do not contain all these parameter information. Therefore, we implemented a vanishing-points-based camera calibration algorithm to calculate the camera matrix and used the Estimation of Distribution Algorithm (EDA) to minimize the reprojection error. In the toolkit the user just needs to manually draw two pairs of parallel lines in the video frame and make each pair orthogonal to the other, then our tool can use these lines to automatically calculate the camera matrix.

C. Object Detection and Tracking

In this step, we use some state-of-art deep learning algorithms to detect and classify all actors in each video frame. Considering the poor video quality in most cases and the accuracy of the DNN model itself, missed detection and incorrect labeling is inevitable. Therefore, the toolkit allows the user to manually edit or correct the automated detection result, as shown in Figure 5.

Next, the toolkit will track all detected actors through the Hungarian algorithm and Kalman filter algorithm, and then extract their trajectory and velocity information in the world coordinate system. The tracking result can also be manually edited. Compared to the 2D bounding box, some key features of the actors can greatly improve data accuracy. For example, the vehicle lamp can be easily detected in HSV color space and it can provide a more accurate location and orientation information than the center of the 2D bounding box. As the next step, we are attempting to infer the 3D bounding box directly using the DNN model.

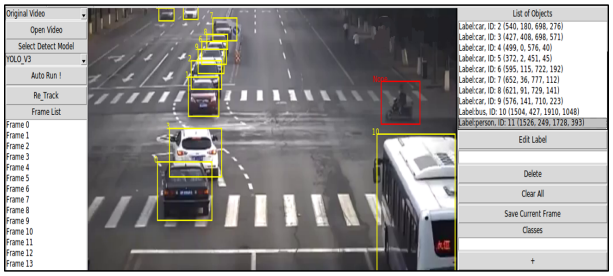

Fig. 5. Object detection and correction. The cars in the yellow bounding box are detected by the DNN module. The missed motorcycle is manually labeled in the red bounding box.

D. Road Network Reconstruction and 3D Rendering

The OpenDRIVE format uses a reference line to define the basic geometry (arcs, straight lines, etc.). Along the reference line, various properties of the road can be defined. Currently in the toolkit, the user needs to manually select sample points to create the reference line, and manually add the road properties, such as lane numbers and connections. With these information, the toolkit can construct the road network and generate the OpenDRIVE format file. Limited by the camera's perspective, in most cases only a portion of the road network is visible. The toolkit will also do road network completion automatically based on the visible portion.

The exported OpenDRIVE format file will be imported into a third-party software RoadRunner [17] for 3D scene rendering and FBX file generation, as shown in Figure 6. The OpenDRIVE file and the FBX file are both used by the graphics engine, e.g. UE4, to generate the 3D simulation environment.

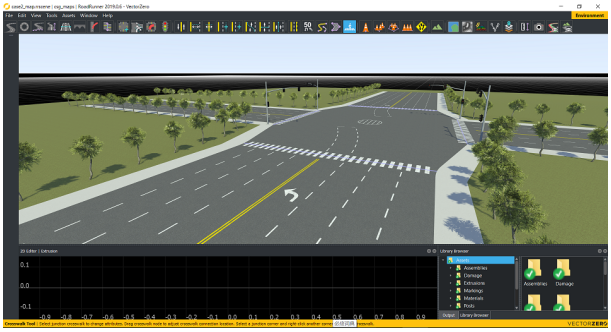

Fig. 6. 3D scene rendering using RoadRunner

E. Scenario Enhancement

Currently we choose two parameters for variation: initial velocity and trajectory. At the beginning of simulation, the other actors, including the troublemaker, will detect the initial velocity of the ego vehicle and then adjust their own initial velocity accordingly. The trajectory of the troublemaker can be regenerated through curve fitting.

F. Simulation and Scoring

Currently the toolkit supports two simulation platforms: CARLA and MATLAB. For CARLA, we basically leverage the existing open source scenario engine of CARLA, scenario runner, but make some modifications to support our scenario specification. MATLAB provides its own functions to read in the scenario specification and OpenDRIVE file, but we need to convert our scenario specification to the MATLAB format.

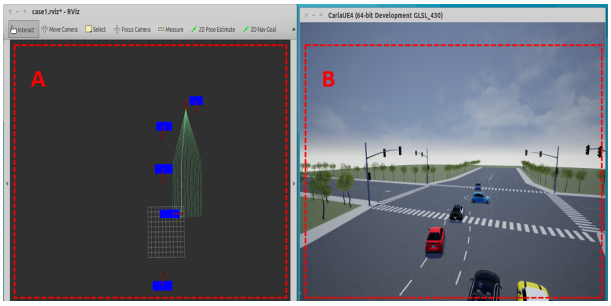

Fig. 7. Simulation on CARLA. A) The ROS visualization tool displaying the available trajectories generated by OpenPlanner. B) View from the ego vehicle's perspective in CARLA.

For demonstrating the usage of the toolkit, we set up an experiment environment to evaluate the OpenPlanner [22] module, which has been implemented in the Autoware [29] open source autonomous driving framework's Robot Operating System (ROS) [19]. The scenario engine rebuilds the scenario in CARLA and sends the simulated perception result to OpenPlanner module through ROS messages. We

can observe the simulation process through multiple monitor windows as shown in Figure 7, together with a test report. We are embedding the Responsibility-Sensitive Safety (RSS) model [20][21] in the test system as an ROS node which can be used for safety scoring.

V. FUTURE WORK

In the future, we will use CSG for multiple studies, e.g., validating and calibrating the RSS model. [9] demonstrated how to utilize 219 safety-critical scenarios in SH-NDS to calibrate the RSS model on autonomous car-following maneuvers, and finally achieve optimal safety performance. They integrated the RSS model into an ACC system, and then used the genetic algorithms (GA) to find the RSS model's optimal parameter combination for car-following maneuvers. With our critical scenario library, we can apply the same method to evaluate the RSS's performance in road intersection environment, and then improve the safety and traffic efficiency of AV passing the road intersection, as shown in Figure 8.

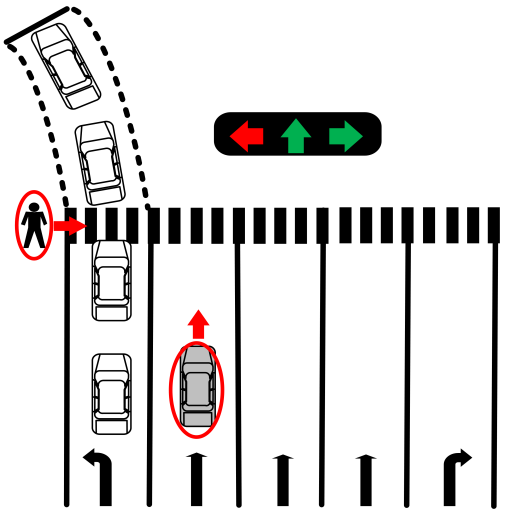

Fig. 8. Critical scenario due to jaywalking. The ego vehicle in gray is going straight through the intersection when the light is green, but meanwhile, a pedestrian is jaywalking through the vehicles waiting for left-turn.

Besides that, another interesting work is to analyze and extract the statistical characteristics of the huge number of scenarios extracted from real accidents, and use them to inspire and conduct the manual design of critical scenarios. We will also explore how to achieve a better interaction between the ego vehicle and other actors in a scenario.

VI. CONCLUSIONS

Scenario simulation has been widely used by AD industry for algorithm development and accelerating AV safety assessment. The difficulty, however, is how to systematically create critical scenarios to cover as many potentially dangerous situations as possible. In the paper we have presented a novel method to generate critical scenarios in a simulator. The scenarios we generate have a very unique value as they are all derived from real traffic accidents. We believe that the development of AV should follow such a path which begins with learning from real-world accidents and then evolves to self-reinforcement learning, just as Google's Alpha Go evolved to Alpha Zero.

The CSG toolkit and the scenario library can be used for multiple purposes. E.g., authorities can use them for AV safety assessment and standard setting, ADS developers can use them for algorithm development and quick regression test before releasing a new version.

ACKNOWLEDGMENT

Thanks for the CARLA team's great effort to implement the rich features required by scenario simulation. Thanks to Ros German, Bernat, and Nstor Subirn, they always responded to our problems quickly and help fix them when we re-build scenarios on CARLA. We also would like to thank Oboril Fabian, Scholl Kay-Ulrich and the Intel Labs Europe team for developing the CARLA scenario engine and sharing their simulation experiences with us. Without their selfless support, we could not complete this study.

REFERENCES

[1] NVIDIA, NVIDIA drive constellation: virtual reality autonomous vehicle simulator (2017).
[2] Shital Shah, Debadeepta Dey, Chris Lovett, and Ashish Kapoor. Airsim:High-fidelity visual and physical simulation for autonomous vehicles. CoRR, abs/1705.05065, 2017.
[3] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. In Proceedings of the 1st Annual Conference on Robot Learning, pages 1C16, 2017.
[4] Queiroz Rodrigo, Berger Thorsten, and Czarnecki Krzysztof. (2019). GeoScenario: An Open DSL for Autonomous Driving Scenario Representation. 287-294. 10.1109/IVS.2019.8814107.
[5] Scenario Runner. https://github.com/carla-simulator/scenario_runner
[6] Driving Scenario Designer. https://www.mathworks.com/help/driving/driving-scenario-simulation.html
[7] Vincenzo Punzo, Maria Teresa Borzacchiello, and Biagio Ciuffo. On the assessment of vehicle trajectory data accuracy and application to the next generation simulation (NGSIM) program data. Transportation Research Part C: Emerging Technologies, 19(6):1243 C 1262, 2011.
[8] Li W, Pan C W, Zhang R, et al. AADS: Augmented autonomous driving simulation using data-driven algorithms[J]. Science Robotics, 2019, 4(28).
[9] Xu Xiaoyan, Wang Xuesong, Wu Xiangbin. Calibration and Evaluation of Responsibility-Sensitive Safety Model on Autonomous Car-Following Maneuvers Using Naturalistic Driving Study Data. 99th Transportation Research Board Annual Meeting.
[10] K. He, G. Gkioxari, P. Dollr, R. Girshick, Mask r-cnn. Proceedings of the 2017 IEEE International Conference on Computer Vision, 2980C2988 (2017).
[11] Krzysztof Czarnecki. Operational world model ontology for automated driving systemsCpart 1: Road structure. Technical report, Waterloo Intelligent Systems Engineering Lab (WISE) Report, 2018.
[12] Till Menzel, Gerrit Bagschik, and Markus Maurer. Scenarios for development, test and validation of automated vehicles. In 2018 IEEE Intelligent Vehicles Symposium, IV 2018, Changshu, Suzhou, China, June 26-30, 2018, pages 1821C1827, 2018.
[13] W. G. Najm, John D. Smith, and Mikio Yanagisawa. Pre-Crash Scenario Typology for Crash Avoidance Research. Technical report, U.S. Department of Transportation, NHTSA, April 2007
[14] Tong D S, AJINKYA B, HERMAN V. Simulation-based Testing Framework for Autonomous Driving Development. 2019 IEEE International Conference on Mechatronics (ICM). New York: IEEE, 2019: 576-583
[15] Heikkila, J., and O. Silven. A Four-step Camera Calibration Procedure with Implicit Image Correction. IEEE International Conference on Computer Vision and Pattern Recognition.1997.

[16] Zhu, M., X. Wang, A. Tarko, and S. Fang. Modeling Car-Following Behavior on Urban Expressways in Shanghai: A Naturalistic Driving Study. 2018.
[17] RoadRunner. https://www.vectorzero.io/
[18] OpenDRIVE. https://www.opendrive.com
[19] Robot Operating System (ROS). https://www.ros.org/
[20] Shalev-Shwartz, S., S. Shammah, and A. Shashua. On a Formal Model of Safe and Scalable Self-Driving Cars. 2017, pp. 1C37.
[21] Shashua, A., S. Shalev-Shwartz, and S. Shammah. Implementing the RSS Model on NHTSA Pre-Crash Scenarios. https://www.mobileye.com/responsibility-sensitive-safety/rss_on_nhtsa.pdf. Accessed Jul. 30, 2019.
[22] Darweesh Hatem, Takeuchi Eijiro, Takeda Kazuya, Ninomiya Yoshiki, Sujiwo Adi, Morales Y., Akai Naoki, Tomizawa Tetsuo, Kato Shinpei. (2017). Open Source Integrated Planner for Autonomous Navigation in Highly Dynamic Environments. Journal of Robotics and Mechatronics. 29. 668-684. 10.20965/jrm. 2017. p0668.
[23] OpenScenario. https://www.asam.net/standards/detail/openscenario
[24] Erdogan, Ahmetcan, et al. "Parametrized end-to-end scenario generation architecture for autonomous vehicles." 2018 6th International Conference on Control Engineering & Information Technology (CEIT). IEEE, 2018.
[25] N. Kalra, S. M. Paddock. Driving to safety: How many miles of driving would it take to demonstrate autonomous vehicle reliability? Transportation Research Part A: Policy and Practice. 94, 182C193 (2016).
[26] A. C. Madrigal, Inside Waymo's Secret World for Training Self-Driving Cars. the Altantis (2017).
[27] Althoff, Matthias, and Sebastian Lutz. "Automatic generation of safety-critical test scenarios for collision avoidance of road vehicles." 2018 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2018.
[28] FBX. https://www.autodesk.com/products/fbx/overview
[29] Autoware. https://www.autoware.ai/
[30] Apollo. https://apollo.auto/
[31] Klischat, Moritz, and Matthias Althoff. "Generating critical test scenarios for automated vehicles with evolutionary algorithms." 2019 IEEE Intelligent Vehicles Symposium (IV). IEEE, 2019.