A Reinforcement Learning Benchmark for Autonomous Driving in Intersection Scenarios

Yuqi Liu[1,2], Qichao Zhang[1,2], Dongbin Zhao[1,2], IEEE Fellow
1.The State Key Laboratory of Management and Control for Complex Systems, Institute of Automation, Chinese Academy of Sciences
2.School of Artificial Intelligence, University of Chinese Academy of Sciences
Beijing, China
liuyuqi2018@ia.ac.cn, zhangqichao2014@ia.ac.cn, dongbin.zhao@ia.ac.cn

2021 IEEE Symposium Series on Computational Intelligence (SSCI) | 978-1-7281-9048-8/21/$31.00 ©2021 IEEE | DOI: 10.1109/SSCI50451.2021.9660172

Abstract—**In recent years, control under urban intersection scenarios has become an emerging research topic. In such scenarios, the autonomous vehicle confronts complicated situations since it must deal with the interaction with social vehicles timely while obeying the traffic rules. Generally, the autonomous vehicle is supposed to avoid collisions while pursuing better efficiency. The existing work fails to provide a framework that emphasizes the integrity of the scenarios while deploying and testing reinforcement learning(RL) methods. Specifically, we propose a benchmark for training and testing RL-based autonomous driving agents in complex intersection scenarios, which is called RL-CIS. Then, a set of baselines consisting various algorithms are deployed. The test benchmark and baselines provide a fair and comprehensive training and testing platform for the study of RL for autonomous driving in the intersection scenario, advancing RL-based methods for autonomous driving control. The code of our proposed framework can be found at https://github.com/liuyuqi123/ComplexUrbanScenarios.**

Index Terms—**autonomous driving, reinforcement learning, intersection scenarios, decision-making**

I. Introduction

Driving under urban scenarios, especially the intersection scenario, is one of the most challenging problems for an autonomous driving(AD) system. According to [1], a general AD system is composed of several subsystems, including sensing, navigation, decision-making, planning, and control. The key challenge of the intersection scenario is the interaction between the autonomous driving vehicle(ADV) and social vehicles, which mainly possess challenges on the decision-making and control modules [2]. Since the behavioral intention of social vehicles is uncertain, the ADV is forced to negotiate and make decisions quickly under solid interactions. Otherwise, traffic accidents are very likely to occur. Generally, the intersection scenario discussed in this paper mainly refers to the ADV passing through a cross-junction while interacting with social vehicles.

Reinforcement learning(RL) methods learn an optimal policy through trial-and-error. An RL agent is able to promote its performance by updating the policy repeatedly. In recent years, RL has become a key technology in the field of AD decision-making [3] and control [4]–[6]. In [7], a method that integrates deep learning (DL) with RL is proposed to address the lane-keeping task on a highway track. And [8], an RL with rule-based constraints method is proposed to train the ADV for the lane-changing task in the highway scenario. In [9], an RL-based planning method is deployed for AD's navigation in an unknown environment.

Besides, some RL-based methods are proposed to deal with the decision-making and control problems in intersection scenarios. In [10], an end-to-end framework is proposed for ADV controlling. The RL agent takes raw data as input and directly outputs the control command of the vehicle. The proposed scenario is mixed up with junctions and roundabouts, while the traffic flow in the scenario is not adjustable. In [11], a multi-agent RL framework is proposed for the behavior model design under intersection scenarios, in which both rule-based and RL-based baselines are provided. The proposed simulator is similar to the one proposed in [12], both of them fail to provide a delicate dynamic model of vehicles and a high-fidelity simulator. In [13], a simulator based on CARLA simulator [14] is proposed, which provides a set of real-world road maps as intersection AD benchmark. Though a partially observable Markov decision process(POMDP) agent is proposed, the RL interface is not deployed for further research. In [15], an RL framework named after ULTRA is proposed with a delicate scenario design. In this paper, the behavior of social vehicles is modeled by the SMARTS simulator [16]. The dynamics of the simulation are idealized as well. We compare the features of some existing frameworks as shown in TABLE I.

As discussed above, previous work has not integrated a tunable intersection scenario benchmark with the RL-based baseline in a high-fidelity simulator. In this paper, we propose the training and testing RL framework addressing the complex intersection scenarios for the AD problem. We call our benchmark Reinforcement Learning Complex Intersection Scenario or RL-CIS for short. The RL-CIS includes two major components. The first is the training environment, in which both scenarios and the RL problem are defined. The second is a set of testing tools, including stochastic and deterministic

This work was supported in part by the National Natural Science Foundation of China (NSFC) under Grants No. 61803371, and also was supported by the Beijing Science and Technology Plan under Grants Z191100007419002 and the Beijing Municipal Natural Science Foundation under Grants L191002

978-1-7281-9048-8/21/$31.00 ©2021 IEEE

test scenarios and performance measurement. Besides, both RL and rule-based methods are provided as the baselines of the proposed framework. As for the original contributions, this paper:

- proposes a benchmark called RL-CIS for RL-based AD agents evaluation. The RL-CIS provides universal environments for testing AD agents, including both stochastic and deterministic test sets,
- develops an RL training environment for intersection scenarios and deploys a traffic flow generation method based on the stochastic process,
- provides basic performance metrics for the AD agent, as well as a set of baselines of both RL and rule-based methods. The experimental results show that our RL agent outperforms some rule-based methods in intersection scenarios.

II. Intersection Scenarios

A. Design of Test Scenarios

In order to evaluate the AD agent, we design a set of intersection scenarios based on [18], which is an industrial standard for ADV field test evaluation. Inspired by [18] and [19], a scenario is defined through a 3-level structure. The first level is the definition of the functional scenario. In this part, the road network structure, target route, behavior of the traffic flow, and other necessary features are defined. The second level is the logical scenario. In this phase, the range of all parameters will be given to constitute the set of all available scenarios. The final stage is the concrete scenario, a specific group of parameters is selected to instantiate a single concrete scenario, which will be rendered for the ADV test.

1) Variability: The variability is the most critical feature for the proposed benchmark. The variability is composed of the diversity of the functional scenarios and the randomness of the logical scenarios, which are the two major rules in scenario design.

a) Diversity: Diversity is emphasized for functional scenarios design. In this phase, the diversity of functional scenarios refers to the variety of the driving tasks and behavior of social vehicles, which leads to a diverse interaction between ego and the social vehicles. In [10]and [13], though the traffic flow is random, the diversity of interaction between ADV and social vehicles is not guaranteed, which means some type of interaction may not happen among the whole test set.

Specifically, in a cross-intersection, the ADV has three potential passing directions, which are turning left, turning right, and going straight. For each driving task, the ego vehicle may encounter interacting traffic flow from different directions. The variation of the interaction is the major diversity of intersection scenarios.

b) Randomness: Randomness is the critical rule in logical scenarios design. In intersection scenarios, randomness mainly indicates two aspects. The first is the randomness of the behavior model of social vehicles. It is defined as that how a social vehicle reacts to potential traffic conflicts. The second aspect is the range of critical kinetic parameters, such as the target speed and minimum brake distance for some rule-based methods. In the design of the range of tunable parameters, there is a trade-off. The parameter range should be as broad as possible to generalize the scenarios, but it should not exceed a reasonable range.

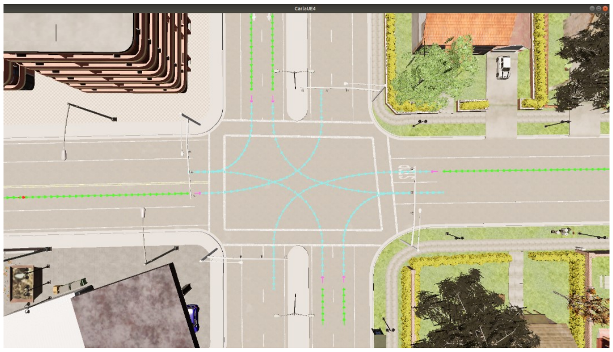


Fig. 1. Intersection scenario in CARLA simulator

2) Deterministic Test Scenarios: The un-signalized intersection scenario is one of the most challenging scenarios for ADV and is mainly discussed in this paper. In the signalized intersection, the traffic lights take over the interaction between vehicles, assuming that all traffic participants obey the traffic rules. Though in real-world traffic, both signalized and un-signalized junctions are common. The un-signalized intersec-

TABLE I
Comparison on Framework of Autonomous Driving in Intersection

Features Frameworks	High-fidelity	Vehicle dynamics	RL Interface	Scenario diversity	Scenario randomness	Evaluation benchmark
CARLA scenario runner [17]	✓	✓	✗	✗	✓	✓
Highway env [12]	✗	✗	✓	✗	✗	✓
interp e2e driving [10]	✓	✓	✓	✗	✗	✗
Summit [13]	✓	✓	✗	✓	✗	✗
BARK [11]	✗	✗	✓	✗	✓	✓
ULTRA [15]	✗	✓	✓	✓	✓	✓
RL-CIS(ours)	✔	✔	✔	✔	✔	✔

tion is selected intentionally to stress the interaction problem in the intersection scenario.

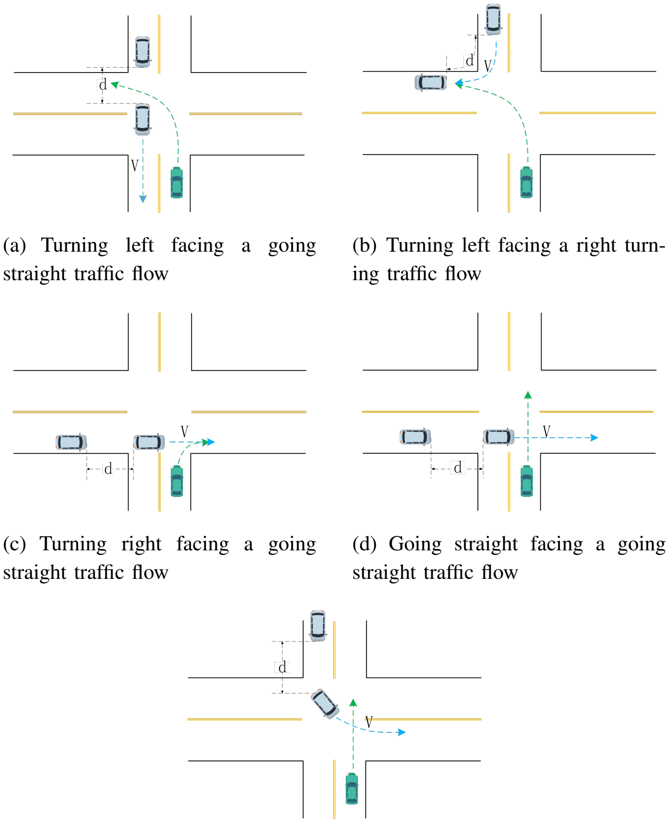
In this paper, we select the cross-intersection scene, which is the most typical road-network structure among intersection scenarios. We use the Town03 map in the CARLA simulator to configure the scenario, as shown in Fig. 1. The blue lines refer to the planned turning routes in the intersection. In a cross-intersection, three typical tasks are mainly considered: turning left, turning right, and going straight. There are multiple potential interacting traffic flows for each turning task that is deterministic by the road network structure. To further decompose scenarios, each functional scenario extracts only a single interacting traffic flow.

Therefore, we propose five functional scenarios as shown in Fig. 2. The functional scenario (a) and (b) refer to the left-turning task, interacting with a going straight and turning right traffic flow coming from the opposite direction. Secondly, scenarios (c) and (d) refer to ego vehicle going straight task, interacting with a going straight and turning left traffic flow coming from left and opposite direction respectively. Thirdly, scenario (e) refers to turning right task interacting with a going straight traffic flow coming from the left direction. The traffic flow in each scenario is composed of a continuous sequence of social vehicles, each vehicle in the traffic flow has a predefined route as illustrated in Fig.1.

The behavior model of all social vehicles is determined by two rules. The first one is speed tracking. The social vehicle will accelerate until it reaches the target speed and then maintain it. The second rule specifies how the social vehicle reacts to the potential conflict, which we employ the autonomous emergency braking(AEB) method. Generally, the AEB model detects a specific range in the front direction, and if any obstacles are detected, the vehicle will hold the brake until the collision detection is clear. Then, the vehicle will continue pursuing the target speed.

Besides the design of functional scenarios, the logical scenario is instantiated with a set of determined kinetic parameters. In our proposed intersection scenarios, the adjustable kinetic parameters are the target speed of each vehicle in the traffic flow V and the gap distance between adjacent vehicles d, as shown in Fig. 2. The gap distance is fixed for each concrete scenario instance to guarantee the stability of the test. The two parameters are determined through the discretization of a specific range. The value of target velocity is sampled from $[10, 40]$km/h uniformly with a step length of 2. The gap distance value is sampled from $[16, 50]$m uniformly with a step length of 2.

3) Stochastic Test Scenarios: Besides the deterministic test, we propose the stochastic test set for the RL-based agent evaluation. In the stochastic test, a comparatively more random traffic flow is provided. Specifically, the behavior model of social vehicles is determined by CARLA's built-in Autopilot function. CARLA Autopilot is a rule-based AD framework that includes navigation, planning, and control modules. The social vehicle controlled by CARLA Autopilot will randomly plan the route to pass through the junction. For the driving behavior, a CARLA autopilot agent has a Boolean switch for collision avoidance with certain vehicles. In our experiment, we switch off the collision avoidance of all social vehicles against RL-based ADV. Because the RL agent learns the policy through trial-and-error, switching off the collision detection will help the RL agent explore and learn policies more efficiently.



(a) Turning left facing a going straight traffic flow

(b) Turning left facing a right turning traffic flow

(c) Turning right facing a going straight traffic flow

(d) Going straight facing a going straight traffic flow

(e) Going straight facing a left turning traffic flow

Fig. 2. Functional scenarios of intersection scenarios

B. Design of Training Scenarios

Generally, the training scenario set is supposed to cover the test scenario set as much as possible but remains some variability at the same time. The traffic flow setting in training scenarios is similar to the one used in test scenarios. We deploy a rule-based traffic flow with adjustable kinetic parameters while each traffic flow holds a fixed route. The kinetic parameters are target speed and gap distance, which are the same as in the deterministic test. However, in the training scenario, the two parameters are various for each vehicle of the same traffic flow. The parameters are sampled from the same interval as defined in the deterministic test when a new social vehicle is spawned. The behavior model of all social vehicles in training scenarios uses the same assumption as the deterministic test, which combines speed tracking and the AEB model.

Therefore, we deploy three agents for turning left, turning right, and going straight tasks, respectively. In each training procedure, the functional scenario with the same task route is trained at the same time. For example, in the left turning task, the RL agent will confront traffic flows shown in Fig.2 (a) and (b) at the same time. The traffic flow on other roads will not be activated.

In the training phase, the RL agent is supposed to explore all available traffic situations fully. As discussed above, the randomness of scenarios is highly affected by the distribution of the kinetic parameters of the social vehicles, which are target speed and gap distance in our proposed test scenarios. Therefore, we deploy a kinetic parameter generation method based on Ornstein–Uhlenbeck(OU) process [20]. The OU process will generate a sequence of kinetic parameters for the whole traffic flow, and each social vehicle of the traffic flow will be given a set of parameters sequentially.

For the kinetic parameters generation, the OU process has two significant advantages. The first is that the cumulative probability distribution of the OU process is Gaussian. The second is that the OU process is mean reverting, limiting the difference between two contiguous sampling values. The stochastic differential equation(SDE) of the OU process is

$$dV_t = \theta \left(\mu - V_t\right) dt + \sigma dW_t \tag{1}$$

in which $\mu, \sigma > 0$, μ refers to the expectation of the variable, $\theta > 0$ denotes the damping factor of the OU process, V_t refers to the target speed of the social vehicle, W_t denotes the Wiener process. The ordinary differential equation(ODE) of the OU process is

$$V_{t+\tau} = \left(1 - e^{-\theta\tau}\right)\mu + V_t e^{-\theta\tau} + \sigma \int_t^{t+\tau} e^{-\theta(\tau - s)} dW_s \tag{2}$$

Since the kinetic parameters of vehicles are bounded by an interval, inspired by [21] we deploy a clipped OU process to avoid over-accumulation on the interval boundary. The clipping process is shown as Algorithm 1. We denote the interval for parameter sampling as $[v_l, v_u]$.

Algorithm 1 Clipped OU process for traffic flow parameter generation

```
Initialize $\hat{V}_{t+\tau}$ with equation (2)
while $\hat{V}_{t+\tau} \notin [v_l, v_u]$ do
    Sample $\hat{V}_{t+\tau}$ from OU process using equation (2)
end while
$V_{t+\tau} = \hat{V}_{t+\tau}$ as the kinetic parameter for next vehicle
```

As for the gap distance between social vehicles, the value of $\hat{V}_t$ is sampled from a truncated gaussian distribution

$$f(d;\mu,\sigma,d_l,d_u) = \begin{cases} 0, & d < d_l \\ \frac{1}{\sigma}\frac{\phi\left(\frac{d-\hat{V}_t}{\sigma}\right)}{\phi\left(\frac{d_u-\hat{V}_t}{\sigma}\right)-\phi\left(\frac{d_l-\hat{V}_t}{\sigma}\right)}, & d_u < d < d_l \\ 0, & d > d_u \end{cases} \tag{3}$$

in which ϕ refers to the normal distribution, $\hat{V}_t$ is the stochastic variable for sampling, d_l, d_u refers to the value interval of the gap distance, σ denotes the variance of the truncated gaussian distribution, which is determined through

$$\sigma = \frac{d_u - d_l}{n} \tag{4}$$

where $n > 0$ is a tunable parameter. It is used to adjust the concentration of the parameters relative to the mean value. According to our proposed sampling method, the gap distance is a linear mapping of target speed over the sampling interval.

III. Baseline Methods

A. Baselines for Intersection Scenarios

1) Rule-based Methods: In this part, we select several classic rule-based methods and an RL algorithm as our proposed baseline methods for the intersection scenarios.

a) Intelligent Driver Model(IDM): The intelligent driver model(IDM) [22] is one of the most popular rule-based baselines for the ADV. The IDM model is designed based on the car-following behavior. The IDM model is defined by the following equations

$$\begin{aligned} \dot{x}_\alpha &= \frac{\mathrm{d}x_\alpha}{\mathrm{d}t} = v_\alpha \\ \dot{v}_\alpha &= \frac{\mathrm{d}v_\alpha}{\mathrm{d}t} = a\left(1 - \left(\frac{v_\alpha}{v_0}\right)^\delta - \left(\frac{s^*\left(v_\alpha, \Delta v_\alpha\right)}{s_\alpha}\right)^2\right) \\ &\text{with } s^*\left(v_\alpha, \Delta v_\alpha\right) = s_0 + v_\alpha T + \frac{v_\alpha \Delta v_\alpha}{2\sqrt{ab}} \end{aligned} \tag{5}$$

where v_0 refers to the desired velocity that the vehicle would drive at in free traffic, s_0 refers to the minimum desired net distance to the car in the front, T refers to the minimum possible time to the vehicle in front, a refers to the maximum vehicle acceleration, $b > 0$ refers to a comfortable braking deceleration, δ usually takes a value of 4.

Besides, the acceleration of vehicle α can be separated into a free road term and an interaction term

$$\begin{aligned} \dot{v}_\alpha^{\text{free}} &= a\left(1 - \left(\frac{v_\alpha}{v_0}\right)^\delta\right) \\ \dot{v}_\alpha^{\text{int}} &= -a\left(\frac{s^*\left(v_\alpha, \Delta v_\alpha\right)}{s_\alpha}\right)^2 \\ &= -a\left(\frac{s_0 + v_\alpha T}{s_\alpha} + \frac{v_\alpha \Delta v_\alpha}{2\sqrt{ab}s_\alpha}\right)^2 \end{aligned} \tag{6}$$

The original IDM method takes the acceleration value as the control output. In our experiments, we deploy a finite difference method to transform the acceleration value to the target speed of the next timestep. The target speed will be mapped to the throttle value of the vehicle by a low-level PID controller.

b) Autonomous Emergency Braking(AEB) Model: The AEB method is a widely used technique of level-2 ADV. In the real-world deployment, the AEB method processes sensor data with a determined algorithm and performs the brake action if any collisions are detected. In the CARLA simulator, we directly use the ground-truth value as the input for deploying the AEB method.

In our work, the AEB method is determined with two rules. The first one is that the longitudinal range of detection L, and the second one is the expansion factor of the bounding box of the social vehicle η. During the driving process, the ego vehicle will detect along its longitudinal direction with a length of L. In the meantime, the bounding box of each social vehicle is expanded from its original physical model according to the expansion factor. More specifically, the size of the bounding box is calculated through

$$L_x = \eta \hat{L_x}, L_y = \eta \hat{L_y}, L_z = \eta \hat{L_z} \tag{7}$$

where L_x, L_y, L_z refer to the original size of social vehicle physical model. If the bounding box of any social vehicle penetrating the front area of the ego vehicle, the AEB model will perform a maximum braking action until the detection area is clear.

2) Reinforcement Learning:

a) State Representation: For the intersection scenario, inspired by [23], we use the ground-truth value of kinetic information of ego vehicle and social vehicles for state representation. Such consideration is common in autonomous driving system design since the major challenge of the intersection scenario comes from the interaction between the ego vehicle and social vehicles. For the ego vehicle, the state vector is defined as $s_e = [v_e, g]$, in which v_e denotes the speed of ego vehicle, and g denotes a 3-dimension one-hot vector, which indicates ego vehicle's current position. For the social vehicle, the state vector of each one is defined as $[v_{i,x}, v_{i,y}, x_i, y_i, cos(\theta_i), sin(\theta_i)]$, in which $v_{i,x}, v_{i,y}$ refer to two-dimensional velocity of social vehicle i, x_i, y_i indicate vehicle's Cartesian coordinates and θ denotes the heading angles under the ego vehicle's coordinate system. The total state representation is combined with the ego vehicle and five nearest social vehicles. All six state vectors are concatenated to a 33-dimension vector as the RL input state vector.

b) Action Space: The action space in our experiments is constructed as a 2-dimension vector $\mathbf{u} = [u_0, u_1]$, which is the output layer of the actor network. Since we use the sigmoid function as the activation function, u_0 and u_1 are limited to $[0, 1]$. Therefore, a trick is deployed for action mapping. The output vector $\mathbf{u}$ is mapped to target speed using $\hat{v}_{ego} = 9 \cdot (u_0 - 0.25 \cdot u_1)$m/s with a numerical truncation constrained to $[0, 9]$m/s. Then the PID controller will map the target speed to the throttle value of the vehicle.

c) Reward Design: Inspired by [24], the reward function is defined through events. More specifically, the reward function is combined with two parts. The first is the reward for each timestep. The second is the final reward at the end of an episode. The complete reward function can be written as follows

$$r = \begin{cases} -0.1 & , \quad t \leq 0.5 \cdot T_{max} \\ +150 & , \quad success \\ -350 & , \quad collision \\ -150 & , \quad time\ over \end{cases} \tag{8}$$

where T_{max} refers to the maximum time limit of one episode, by which common sub-task reward encourages ego vehicle to improve traffic efficiency. In our experiments, the termination condition is defined through these events. The success condition refers to that the ego vehicle reaches the termination waypoint without collision in the given time. The collision and time-over conditions indicate the failure result. Both success and failure conditions will directly lead to the termination of an episode.

d) RL Algorithms: For the intersection scenario, we deploy the TD3 algorithm [25] as our RL baseline. For the neural network design, the state vector is divided by ego and social vehicle. Each component is followed by an encoder network, which is formed by 64×64 fully connected(FC) layers. The output of encoders is concatenated and followed by an FC layer. The actor and critic network of the TD3 algorithm share the same network structure in our experiments. For the critic network, the FC layers are connected by the tanh function. The actor network uses the same activation function except for the output layer, which is the sigmoid function.

IV. Simulation Experiments

A. Evaluation Metrics

Many metrics can be used to measure the behavior of agents [26]. For an AD system, safety and efficiency are the most concerned performance index. In our framework, success rate and average passing time are general metrics for performance evaluation. Success rate indicates that how the RL agent performs for the specified task directly. In this paper, the success rate is defined as

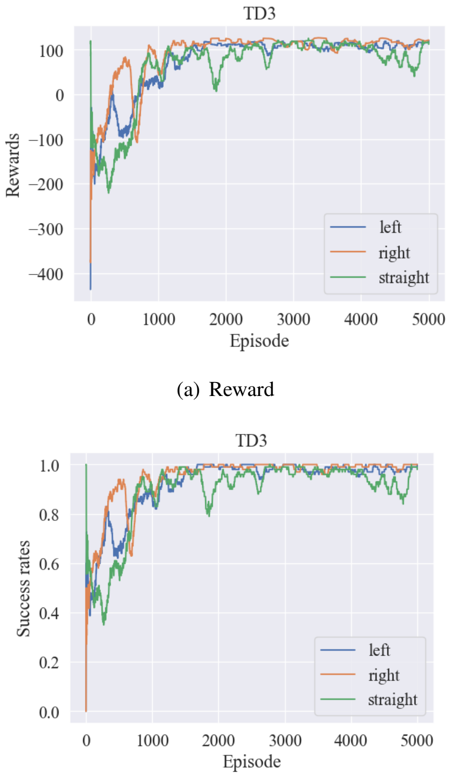
$$SuccessRate = \frac{SuccessCounts}{TotalTestNumber} \times 100\% \tag{9}$$

Secondly, efficiency is measured through the average duration time of a single testing episode. Since the functional scenario which targets on same turning task shares the same route, the average passing time is compared through route classification. It is important to note that we only count the time of the successful test. All scenarios are divided into three groups for average passing time comparison.

B. Results and Analysis

1) Training Process: The learning curves of RL baselines in the task of highway training scenario are depicted in

Fig. 3. The figure shows that the RL agent for each task converges fast within 2000 episodes. Besides, for each route task, the learning curve converges to a maximum level within 5000 episodes. Then the left turning and right turning agents maintain relatively high and stable performance while the going straight agent has slight fluctuation. The main reason for such appearance is because that the agent in going straight task must interact with the left turning traffic flow coming from the opposite direction, which brings greater challenge than other tasks.



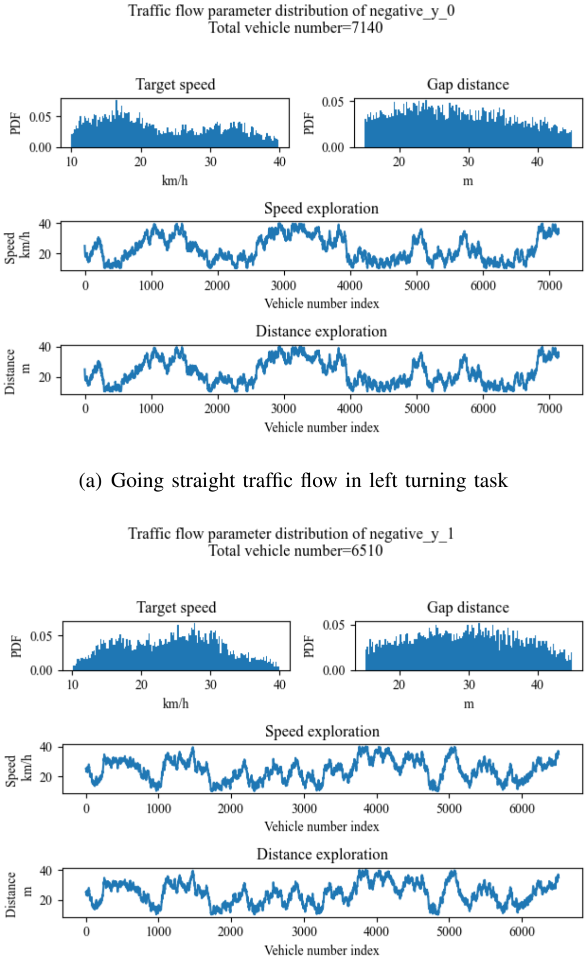
(a) Reward

(b) Success rate

Fig. 3. Intersection training curve

In the training process of each task, the interacting traffic flows are activated according to the definition of logical scenarios, as described in experiment settings. The kinetic parameters generation of each traffic flow is counted, the sampling and distribution in the training of the left-turning task are shown in Fig.4. The distribution of parameters sampling is approximate to Gaussian distribution, while the sampling curve of the time domain is rather smooth.

2) Deterministic Test: In the deterministic test, we deploy a TD3 agent as the RL baseline. Besides, two rule-based methods are deployed as a comparison, which are IDM and the AEB model. In our deployment in CARLA, the IDM model will detect a certain distance along the driving route, pursuing either speed tracking or car following behavior as shown in (5) and (6). Compared to the AEB method, the IDM agent adjusts its velocity rather smoothly. The experimental results are shown in TABLE II. We evaluate the TD3 agent and rule-based agents in all five functional scenarios. The statistics are calculated by the task routes for the rules-based agents since they poor performance relatively. In turning left and turning right experiments, the RL agent reaches a success near 90%, and exceeds the rules-based agent in both success rate and average time. According to the result, going straight is the most challenging task since the left-turning social vehicles are rather fast and hard to negotiate for the ego vehicle.



(a) Going straight traffic flow in left turning task

(b) Turning right traffic flow in left turning task

Fig. 4. OU process-based kinetic parameters sampling and distribution

3) Stochastic Test: The experimental results of the stochastic test are shown in Table III. In the stochastic test, the kinetic parameters of traffic flows are uniformly sampled from an interval, which makes traffic flows not extremely dense. Therefore the RL agent reaches a significantly higher success rate for each task compared to the deterministic test.

In this part, the RL method is compared with rule-based methods as well. As the table shows, the RL agent outperforms rule-based methods in both success rate and average time. The overall success rate of the TD3 agent is above 90%. Though in turning left task and going straight task, rule-base agents have better the average time, they are at a definite disadvantage in

TABLE II
DETERMINISTIC TEST RESULTS FOR INTERSECTION SCENARIOS

Functional scenario		(a)	(b)	(c)	(d)	(e)
Ego route		Turning left		Turning right	Going straight	
TD3	Success rate(%)↑	**94.8**	**93.8**	**89.24**	**99.0**	**80.0**
	Average time(s)↓	**6.83**	**6.65**	**7.04**	**6.94**	**4.79**
IDM	Success rate(%)↑	67.7		62.8	47.4	
	Average time(s)↓	8.40		8.23	8.59	
AEB	success rate(%)↑	72.74		50.0	48.96	
	average time(s)↓	8.66		7.21	7.18	

terms of safety. We believe such results occur because the IDM and AEB methods have a limited input, which makes them not capable of detecting potential conflict with social vehicles from the cross direction. In conclusion, the model-free RL agent dominates our proposed benchmark of the intersection scenarios.

TABLE III
STOCHASTIC TEST RESULTS FOR INTERSECTION SCENARIOS

Methods	Driving task	Success rate(%)↑	Average time(s)↓
TD3(ours)	Left	**95.9**	9.78
	Right	**96.9**	**7.64**
	Straight	**91.9**	9.20
IDM	Left	68.3	11.21
	Right	72.7	11.05
	Straight	33.7	22.6
AEB	Left	71.3	**9.06**
	Right	88.7	8.86
	Straight	58.3	**8.83**

V. CONCLUSION

In this paper, we propose RL-CIS as a framework to train and test the RL-based AD agent in intersection scenarios. Firstly, a group of functional scenarios for un-signalized intersection is designed. Then the behavioral model of social vehicles is determined with two critical parameters, composed of the whole logical scenario set. The concrete scenarios are proposed by discretizing logical scenarios and become the deterministic test set in our proposed framework. Besides, we deploy a set of stochastic tests to evaluate the RL-based AD agent further. Meanwhile, the training environment for the RL agent is developed. In this part, a stochastic process-based sampling method is deployed for traffic flow parameters generation. Both the training and test sets are built through the CARLA simulator. In addition to that, we offer a set of baselines for the intersection AD benchmarks, including TD3, IDM, and AEB methods. According to the experimental results, the RL agent shows significant superiority compared with rule-based methods.

REFERENCES

[1] E. Yurtsever, J. Lambert, A. Carballo, and K. Takeda, “A survey of autonomous driving: Common practices and emerging technologies,” *IEEE access*, vol. 8, pp. 58 443–58 469, 2020.

[2] B. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. Sallab, S. Yogamani, and P. Pérez, “Deep reinforcement learning for autonomous driving: A survey. arxiv 2020,” *arXiv preprint arXiv:2002.00444*.

[3] D. Li, D. Zhao, and Q. Zhang, “Reinforcement learning based lane change decision-making with imaginary sampling,” in *2019 IEEE Symposium Series on Computational Intelligence (SSCI)*. IEEE, 2019, pp. 16–21.

[4] B. Wang, D. Zhao, and J. Cheng, “Adaptive cruise control via adaptive dynamic programming with experience replay,” *Soft Computing*, vol. 23, no. 12, pp. 4131–4144, 2019.

[5] D. Zhao, Z. Hu, Z. Xia, C. Alippi, Y. Zhu, and D. Wang, “Full-range adaptive cruise control based on supervised adaptive dynamic programming,” *Neurocomputing*, vol. 125, pp. 57–67, 2014.

[6] D. Zhao, Z. Xia, and Q. Zhang, “Model-free optimal control based intelligent cruise control with hardware-in-the-loop demonstration [research frontier],” *IEEE Computational Intelligence Magazine*, vol. 12, no. 2, pp. 56–69, 2017.

[7] D. Li, D. Zhao, Q. Zhang, and Y. Chen, “Reinforcement learning and deep learning based lateral control for autonomous driving,” *IEEE Computational Intelligence Magazine*, vol. 14, no. 2, pp. 83–98, 2019.

[8] J. Wang, Q. Zhang, D. Zhao, and Y. Chen, “Lane change decision-making through deep reinforcement learning with rule-based constraints,” in *2019 International Joint Conference on Neural Networks*. IEEE, 2019, pp. 1–6.

[9] H. Li, Q. Zhang, and D. Zhao, “Deep reinforcement learning-based automatic exploration for navigation in unknown environment,” *IEEE transactions on neural networks and learning systems*, vol. 31, no. 6, pp. 2064–2076, 2019.

[10] J. Chen, S. E. Li, and M. Tomizuka, “Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning,” *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[11] J. Bernhard, K. Esterle, P. Hart, and T. Kessler, “Bark: Open behavior benchmarking in multi-agent environments,” in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2020. [Online]. Available: https://arxiv.org/pdf/2003.02604.pdf

[12] E. Leurent, “An environment for autonomous driving decision-making,” https://github.com/eleurent/highway-env, 2018.

[13] P. Cai, Y. Lee, Y. Luo, and D. Hsu, “Summit: A simulator for urban driving in massive mixed traffic,” in *2020 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2020, pp. 4023–4029.

[14] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun, “CARLA: An open urban driving simulator,” in *Proceedings of the 1st Annual Conference on Robot Learning*, 2017, pp. 1–16.

[15] M. Elsayed, K. Hassanzadeh, N. M. Nguyen, M. Alban, X. Zhu, D. Graves, and J. Luo, “Ultra: A reinforcement learning generalization benchmark for autonomous driving,” 2020. [Online]. Available: https://ml4ad.github.io/files/papers2020/ULTRA:Areinforcementlearninggeneralizationbenchmarkforautonomousdriving.pdf

[16] M. Zhou, J. Luo, J. Villella, Y. Yang, D. Rusu, J. Miao, W. Zhang, M. Alban, I. Fadakar, Z. Chen *et al.*, “Smarts: Scalable multi-agent reinforcement learning training school for autonomous driving,” *arXiv preprint arXiv:2010.09776*, 2020.

[17] CARLA Team, “Scenariorunner for carla,” 2021. [Online]. Available: https://https://github.com/carla-simulator/scenario_runner

[18] China ITS Industry Alliance. (2020) Requirements of simulation scenario set for automated driving vehicle. [Online]. Available: http://www.ttbz.org.cn/StandardManage/Detail/38842/

[19] E. Deelman, K. Vahi, G. Juve, M. Rynge, S. Callaghan, P. J. Maechling, R. Mayani, W. Chen, R. Ferreira da Silva, M. Livny, and K. Wenger, "Pegasus: a workflow management system for science automation," *Future Generation Computer Systems*, vol. 46, pp. 17–35, 2015, funding Acknowledgements: NSF ACI SDCI 0722019, NSF ACI SI2-SSI 1148515 and NSF OCI-1053575. [Online]. Available: http://pegasus.isi.edu/publications/2014/2014-fgcs-deelman.pdf

[20] S. Finch, "Ornstein-uhlenbeck process," 2004.

[21] G. Huang, M. Mandjes, and P. Spreij, "Limit theorems for reflected ornstein–uhlenbeck processes," *Statistica Neerlandica*, vol. 68, no. 1, pp. 25–42, 2014.

[22] A. Kesting, M. Treiber, and D. Helbing, "Enhanced intelligent driver model to access the impact of driving strategies on traffic capacity," *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 368, no. 1928, pp. 4585–4605, 2010.

[23] E. Leurent and J. Mercat, "Social attention for autonomous decision-making in dense traffic," *arXiv preprint arXiv:1911.12250*, 2019.

[24] T. Tram, A. Jansson, R. Grönberg, M. Ali, and J. Sjöberg, "Learning negotiating behavior between cars in intersections using deep q-learning," in *2018 21st International Conference on Intelligent Transportation Systems*. IEEE, 2018, pp. 3169–3174.

[25] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*. PMLR, 2018, pp. 1587–1596.

[26] A. Tampuu, T. Matiisen, M. Semikin, D. Fishman, and N. Muhammad, "A survey of end-to-end driving: Architectures and training methods," *IEEE Transactions on Neural Networks and Learning Systems*, 2020.