Comprehensive Training and Evaluation on Deep Reinforcement Learning for Automated Driving in Various Simulated Driving Maneuvers

Dong, Yongqi; Datema, Tobias; Wassenaar, Vincent; van de Weg, Joris; Kopar, Cahit Tolga; Suleman, Harim

DOI
10.1109/ITSC57777.2023.10422159

Publication date
2023

Document Version
Final published version

Published in
2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC)

2023 IEEE 26th International Conference on
Intelligent Transportation Systems (ITSC)
24-28 September 2023. Bilbao, Bizkaia, Spain

Comprehensive Training and Evaluation on Deep Reinforcement Learning for Automated Driving in Various Simulated Driving Maneuvers

Yongqi Dong, Tobias Datema, Vincent Wassenaar, Joris van de Weg, Cahit Tolga Kopar, and Harim Suleman

2023 IEEE 26th International Conference on Intelligent Transportation Systems (ITSC) | 979-8-3503-9946-2/23/$31.00 ©2023 IEEE | DOI: 10.1109/ITSC57777.2023.10422159

***Abstract*— Developing and testing automated driving models in the real world might be challenging and even dangerous, while simulation can help with this, especially for challenging maneuvers. Deep reinforcement learning (DRL) has the potential to tackle complex decision-making and controlling tasks through learning and interacting with the environment, thus it is suitable for developing automated driving while not being explored in detail yet. This study carried out a comprehensive study by implementing, evaluating, and comparing the two DRL algorithms, Deep Q-networks (DQN) and Trust Region Policy Optimization (TRPO), for training automated driving on the *highway-env* simulation platform. Effective and customized reward functions were developed and the implemented algorithms were evaluated in terms of on-lane accuracy (how well the car drives on the road within the lane), efficiency (how fast the car drives), safety (how likely the car is to crash into obstacles), and comfort (how much the car makes jerks, e.g., suddenly accelerates or brakes). Results show that the TRPO-based models with modified reward functions delivered the best performance in most cases. Furthermore, to train a uniform driving model that can tackle various driving maneuvers besides the specific ones, this study expanded the *highway-env* and developed an extra customized training environment, namely, *ComplexRoads*, integrating various driving maneuvers and multiple road scenarios together. Models trained on the designed *ComplexRoads* environment can adapt well to other driving maneuvers with promising overall performance. Lastly, several functionalities were added to the *highway-env* to implement this work. The codes are open on GitHub at https://github.com/alaineman/drlcarsim-paper.**

I. INTRODUCTION

Artificial intelligence (AI) is making huge improvements in various fields, one of which is automated driving [1]. One typical type of AI that is well-suitable for developing automated driving models is Deep Reinforcement Learning (DRL) [2]. DRL makes use of the advantage of deep neural networks regarding feature extraction and the advantage of reinforcement learning regarding learning from interacting with the environment. DRL exhibits excellent performance in various decision-making tasks, e.g., *GO* [3] and playing video games [4] and it has been employed in various automated driving tasks [5]–[7], e.g., lane-keeping, lane-changing, overtaking, ramp merging, and driving through intersections.

For the lane-keeping task, Sallab et al. [8], [9] developed DRL-based methods for delivering both discrete policies using Deep Q-network (DQN) and continuous policies using Deep Deterministic Actor-Critic Algorithm (DDAC) to follow the lane and to maximize the average velocity when driving on the curved race track on Open Racing Car Simulator (TORCS). Similarly, for the lane-changing task, Wang et al. [10] trained a DQN-based model to perform decision-making of lane-keeping, lane changing to the left/right, and acceleration/deceleration, so that the trained agent can intelligently make a lane change under diverse and even unforeseen scenarios. Furthermore, Zhang et al. [11] proposed a bi-level lane-change behavior planning strategy using DRL-based lane-change decision-making model and negotiation-based right-of-way assignment model to deliver multi-agent lane-change maneuvers. For the overtaking task, Kaushik et al. [12] adopted Deep Deterministic Policy Gradients (DDPG) to learn overtaking maneuvers for an automated vehicle in the presence of multiple surrounding cars in a simulated highway scenario. They verified that their curriculum learning resembled approach can learn to smooth overtaking maneuvers, largely collision-free, and independent of the track and number of cars in the scene. For the ramp merging task, Wang and Chan [13] employed a Long-Short Term Memory (LSTM) neural network to model the interactive environment conveying internal states containing historical driving information to a DQN which then generated Q-values for action selection regarding on-ramp merging. Additionally, for negotiating and driving through intersections, Isele et al. [14] explored the effectiveness of the DQN-based DRL method to handle the task of navigating through unsignaled intersections. Finally, Guo and Ma [15] developed a real-time learning and control framework for signalized intersection management, which integrated both vehicle trajectory control and signal optimization using DDPG-based DRL learning directly from the dynamic interactions between vehicles, traffic signal control and traffic environment in the mixed connected and automated vehicle (CAV) environment.

It is observed that although many studies have utilized DRL for various driving tasks, most of them focus only on one specific driving maneuver. Seldom do they evaluate the DRL model performance across different maneuvers and neither do they explore the adaptability of DRL models

*This work is supported by Applied and Technical Sciences (TTW), a subdomain of the Dutch Institute for Scientific Research (NWO), Grant/Award Number: 17187

Yongqi Dong is with the Department of Transport and Planning, Delft University of Technology, Delft, 2628 CN, the Netherlands (email: y.dong-4@tudelft.nl).

Tobias Datema, Vincent Wassenaar, Joris van de Weg, Cahit Tolga Kopar, and Harim Suleman are with the Faculty of Electrical Engineering, Mathematics & Computer Science, Delft University of Technology, Delft, 2628 CN, the Netherlands (email: {T.Datema, V.Wassenaar, J.J.vandeWeg, C.T.Kopar, H.I.Suleman}@student.tudelft.nl).

trained on one specific environment but tested in other various maneuvers. This study tries to fill this research gap by implementing, evaluating, and comprehensively comparing the performance of two DRLs, i.e., DQN and TRPO, in various driving scenarios. Customized effective reward functions were developed and the implemented DRLs were evaluated in terms of various aspects considering driving safety, efficiency, and comfort level. This study also constructed a new simulation environment, named *'ComplexRoads'* (shown in Fig 1), integrating various driving maneuvers and multiple road scenarios. The *ComplexRoads* served to train a uniform driving model that can tackle various driving tasks. And to verify this, the models trained only on *ComplexRoads* were tested and evaluated in the specific driving maneuvers. Intensive experimental results demonstrated the effectiveness of this customized training environment.

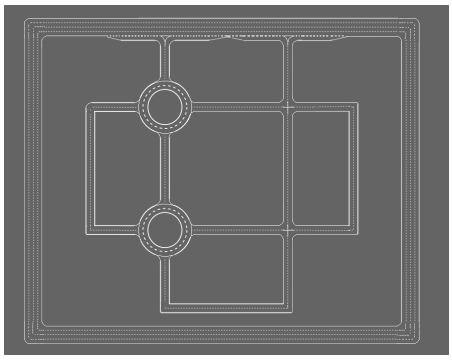


Fig. 1: The layout of *ComplexRoads* environment

To advance the learning capability for the developed DRL-based AI models, i.e. encouraging relational insight, besides designing *ComplexRoads*, several built-in functions of the `highway-env` package were also upgraded. Notable modifications are summarized as follows: the tracking of the 'current' lane with respect to the car (training agent) was upgraded to take into account the lane heading to eliminate confusing transitions when driving off-road. Furthermore, the distance between the car and its current lane was upgraded to a signed value to allow for orientation distinction. Similarly, the lane heading difference, LHD for short, was adjusted to also be a signed value. These improvements yield increased learning abilities for both on-road driving, returning to on-road driving when off-road, and a general sense of 'awareness' given an arbitrary environment.

II. METHODOLOGY

A. System Framework

The general DRL learning cycle is an iterative learning process based on the agent's performance in the environment influenced by the agent's actions. In mathematical terms, automated driving can be modeled as a Markov Decision Process (MDP) [16]. MDP captures the features of sequential decision-making. The components of an MDP include environments, agents, actions, rewards, and states. In this study, the system framework which illustrates the corresponding MDP is depicted in Fig 2. The system generally consists of five main elements, i.e., environment, agent, action, state, and reward, which will be elaborated in detail in this section.

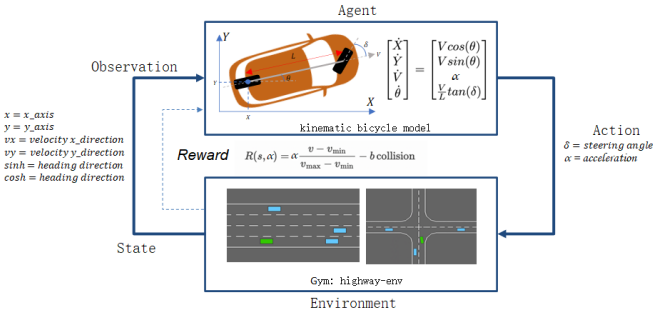


Fig. 2: The system framework-illustration of the DRL MDP.

B. DRL MDP Elements

Environment: To simulate the MDP, this study adopted the *highway-env* platform [17], which is a Python-based package that offers a variety of driving environments. As a widely used platform, ample research has been conducted using the *highway-env*, such as [18], [19]. In the *highway-env*, six dedicated driving scenarios are available, i.e., Highway, Merge, Roundabout, Intersection, Racetrack, and Parking. Users can also customize environments by specifying the number of lanes, the size of a given roundabout, and other parameters. In this study, all the driving scenarios, except for the Highway and Parking, are covered.

For training and evaluating a uniform driving model, this study designed a new simulation environment, named *'ComplexRoads'* (shown in Fig 1). *'ComplexRoads'* integrates two highway merging scenarios, two four-way intersections, two roundabouts, and several segments of multi-straight lanes. The DRL models trained only on *ComplexRoads* were tested and evaluated in the specific driving maneuvers originally available on *highway-env*.

Agent: A kinematic bicycle model is used to represent the vehicle as the agent of MDP. Despite its simplicity, a kinematic bicycle model is able to represent actual vehicle dynamics [20].

Action: An action taken by the agent in the proposed MDP is an element from the contracted *Action Space*. In this study, the two dimensions of the Action Space $\mathcal{A}$ are: acceleration (throttle) and steering angle of the front wheels. Depending on the DRL algorithm $\mathcal{A}$ is either of the form $\left[-\frac{\pi}{2}, \frac{\pi}{2}\right] \times [-5, 5]$ for algorithms requiring a continuous action space, or $\{\delta_1, \ldots, \delta_n\} \times \{\alpha_1, \ldots \alpha_m\}$ in the $n \times m$ discrete case. Hence, $(\delta, \alpha) \in \mathcal{A}$, where steering is denoted by δ and acceleration is denoted by α.

State: As illustrated in Fig 2, the state in the proposed MDP includes the ego AV's state, e.g., location (x, y), velocity (v_x, v_y), and heading direction, together with the surrounding vehicles state and road conditions and is directly accessible at each time frame to the ego car, either in absolute terms or relative to itself.

Reward: The customized *Reward* function is elaborated in detail in the following subsection *C*.

C. Reward Function

For training the models, this study used the reward function already present in the *highway-env* package (referred to as the baseline reward and is illustrated in the middle of Fig 2) and the own modified and upgraded reward function. The model performances were compared to demonstrate that the upgraded reward is better than the baseline reward. During the training it is observed that in the early stages, the trained agent car would sometimes drive off the road. To make the training more efficient in handling the off-road driving and stimulating the agent to return to driving on-road, one specific contribution in this study is to adjust the distance measure between the agent and the lane, in addition with constructing the lane heading difference measure illustrated in the following paragraphs.

Let c denote the ego car agent and $\mathcal{L}$ the corresponding lane. A lane is a collection of lane points $l \in \mathcal{L}$. Now define l' as the lane point with the shortest Euclidean distance to the car, meaning

$$l' := \arg\min_{l \in \mathcal{L}} d(c, l') \tag{1}$$

and define the orientation ω of the car c with respect to a lane point l as follows

$$\omega(c, l) = \begin{cases} 1 & \text{if car is located left of } l^1 \\ -1 & \text{otherwise} \end{cases} \tag{2}$$

Then, this study defines the distance between the ego car and the lane as the shortest distance from the ego car c, to any point l on lane $\mathcal{L}$, meaning

$$d(c, \mathcal{L}) = \omega(c, l')d(c, l') \tag{3}$$

The car heading and lane point heading are denoted by c_φ and l_φ respectively, both values are within angle range $(-\pi, \pi]$. Now, the lane heading difference (LHD) is defined as

$$\text{LHD} = \begin{cases} l_\varphi - c_\varphi + 2\pi & \text{if } l_\varphi - c_\varphi < -\pi \\ l_\varphi - c_\varphi - 2\pi & \text{if } l_\varphi - c_\varphi > \pi \\ l_\varphi - c_\varphi & \text{otherwise} \end{cases} \tag{4}$$

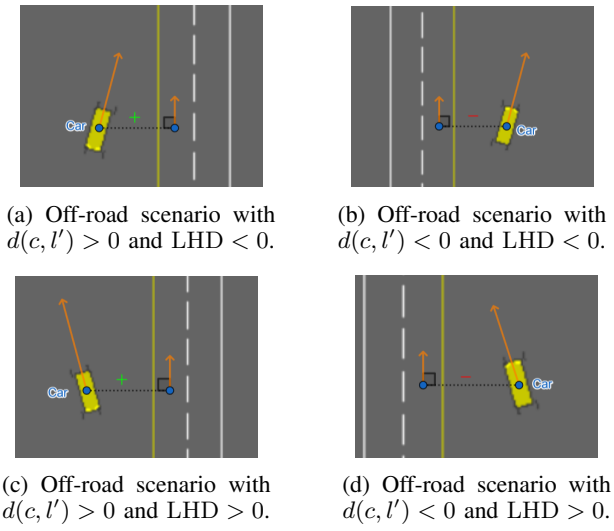
An important remark to this setup is the fact that if $\text{sgn}(\text{LHD}) \cdot \text{sgn}(d(c, \mathcal{L})) < 0$ then the car is heading for the lane. Similarly, if $\text{sgn}(\text{LHD}) \cdot \text{sgn}(d(c, \mathcal{L})) > 0$ the car is deviating (further) from the lane.

Finally, denote the velocity of the ego car c by c_v, the reward function $R : \mathbb{R}^3 \to \mathbb{R}$, with regard to *state* S, is defined as

$$R_S(c, \mathcal{L}) = \begin{cases} \frac{\cos(|\text{LHD}|) \cdot c_v}{20 \cdot \max(1, |d(c, \mathcal{L})|)} & \text{if } c_v \geq 0 \\ 0 & \text{otherwise} \end{cases} \tag{5}$$

where LHD is the lane heading difference between the ego car and the closest lane point. However, if the car crashes during the simulation, the reward is automatically set as -10, regardless of the *state*.

[1]More precisely; if the car is located left of the tangent line for the lane segment containing l



(a) Off-road scenario with $d(c, l') > 0$ and LHD < 0.

(b) Off-road scenario with $d(c, l') < 0$ and LHD < 0.

(c) Off-road scenario with $d(c, l') > 0$ and LHD > 0.

(d) Off-road scenario with $d(c, l') < 0$ and LHD > 0.

Fig. 3: Four different off-road scenarios showcasing available environment observations of the ego car. Both lane heading and car heading are portrayed by vectors. The lane distance and LHD, for the ego car c with respect to the lane point l'. The sign is orientation based: if the car is located left of the road, the Euclidean distance is perceived as positive, and negative if located right of the road.

The reward function, as defined in Equation 5, rewards the car for its 'effective' speed on the road, defined by the cosine of the angular difference between the direction the car is driving in and the direction in which the road goes, multiplied by the speed of the car. With this design, both an increase in the driving speed and driving in line with the road heading will result in high rewards. Moreover, the value is divided by the lane offset to punish the car for driving off-road and also divided by 20 to scale the reward function to remain close to 1 under optimal circumstances.

D. DRL Algorithms

Regarding DRL algorithms, TRPO [21] and DQN [22] were customized and implemented. Details of the DRLs including hyperparameter settings are elaborated in the supplementary at https://shorturl.at/oLP57, while Section IV presents the results comparing trained DRLs' performances.

E. Evaluation of the Models

To evaluate and compare the model performance, one needs a set of indicators and metrics. For which this study implemented a performance logger that measures and stores various indicators when testing a model in a given environment. These indicators are measured for a set amount of runs and the logger then prints the average values over all the runs. The measured indicators are: 1) Speed, 2) Peak jerk, 3) Total jerk, 4) Total distance, 5) Total steering, 6) Running time, 7) Lane time (rate of time the car is running within the road), and 8) Rate of collision.

The jerk is defined as the difference between the current and the previous action of a vehicle, consisting of both the steering angle and the acceleration. The magnitude of the total jerk reflects the degree to which the vehicle's motion

changes abruptly and frequently, where a higher value of the total jerk implies a less comfortable driving. The jerk is defined by equations in 6:

$$J_{\text{acceleration}} = \frac{a_{t-1} - a_t}{a_{\max} - a_{\min}}$$
$$J_{\text{steering}} = \frac{w_t - w_{t-1}}{w_{\max} - w_{\min}} \quad (6)$$
$$J_{\text{total}} = \frac{J_{\text{acceleration}} + J_{\text{steering}}}{2}$$

The total steering is defined as the total sum of steering the car performs in the course of an evaluation, measured in angles. A higher amount of steering could, to certain extent, imply less efficient driving with unnecessary steering.

The onlane rate is defined as the amount of time the evaluated car spends driving on the lane, divided by the total amount of time the car spends driving. The collision rate is defined as the total amount of collisions the car makes, divided by the total amount of evaluation trials.

III. Experiments

This study conducted intensive experiments to train and evaluate DRL models using TRPO and DQN algorithms on four environments provided by *highway-env*, and also the newly self-designed *ComplexRoads*. The models were trained using both the original standard reward function provided by *highway-env* (which served as the baseline) and the customized reward function. The hyperparameters used for training can be found in the appendix at https://shorturl.at/oLP57. The models were trained on the supercomputer Delft Blue [23]. For every environment, ten models were trained and saved for 10,000 and 100,000 iterations. When finishing training, the model performance was tested for 10 runs. During the performance testing, constraints such as a maximum running time, minimum speed and if a crash had occurred were adopted. To obtain an overall assessment, the average of all these 10 testing results was calculated. To get an idea of how well the models perform regarding an uniform driving model, they were not only tested in their trained environments, but also cross-evaluated in other different environments. With the cross-evaluating, the effectiveness of the newly designed environment *ComplexRoads* can be verified. The experiment testing results are summarized and discussed in Section IV.

IV. Results and Discussion

Tables I, II, III, IV, and V present the average performances of the DRL models trained on five environments and evaluated on the same respective environment. For every model variant in one specific environment, this study trained it for 10 times and also evaluated it for 10 times to get the average performance indicators. This paper writes "1*" when the number is rounded to 1, but not quite equal to 1. With the letters "B" and "M", this paper refers to whether the baseline reward function or modified reward function was used in training the model.

Meanwhile, Tables VI, VII, VIII, IX, XI, and X present the average performances of the implemented DRL models trained in their own environment, but evaluated in other different environments. This is for evaluating how adaptive these models are. In order to save space, these tables leave out some of the 'less important' indicators, which can be found in the appendix at https://shorturl.at/oLP57.

One needs to note that for the environment of Merge and the self-designed *ComplexRoads*, no baseline reward functions are available, so only the models trained by the modified and upgraded reward (indicated with "-M") were evaluated. Also, for cross-environments evaluating, only models with the modified reward were evaluated.

TABLE I: *ComplexRoads*

Indicator	DQN-M	TRPO-M
speed	16.1	16.2
pk. jerk	0.99	0.799
tot. jerk	221	13.2
tot. distance	661	547
tot. steering	263	46.3
runtime	607	492
onlane rate	0.999	0.999
col. rate	0.07	0.09

TABLE II: Roundabout

Indicator	DQN-B	DQN-M	TRPO-B	TRPO-M
speed	8.3	8.6	7.88	8.25
pk. jerk	1.07	1.09	0.704	0.775
tot. jerk	71	159	15.4	20.7
tot. distance	185	278	214	229
tot. steering	128	210	92.8	114
runtime	318	479	382	394
onlane rate	0.384	0.783	0.341	0.693
col. rate	0.71	0.68	0.51	0.62

TABLE III: Intersection

Indicator	DQN-B	DQN-M	TRPO-B	TRPO-M
speed	9.89	10.1	9.74	10.3
pk. jerk	0.892	1.04	0.545	0.637
tot. jerk	24.3	32.6	6.21	6.53
tot. distance	38.6	62.8	65	68.3
tot. steering	29.9	41.4	18.7	18.5
runtime	58.8	93.3	101	100
onlane rate	0.988	0.999	0.999	1*
col. rate	0.38	0.49	0.33	0.19

TABLE IV: Merge

Indicator	DQN-M	TRPO-M
speed	30.9	29.1
pk. jerk	0.863	0.607
tot. jerk	47.8	11.2
tot. distance	491	487
tot. steering	86.7	82.1
runtime	226	253
onlane rate	0.875	0.836
col. rate	0.5	0.4

TABLE V: Racetrack

Indicator	DQN-B	DQN-M	TRPO-B	TRPO-M
speed	7.12	9.44	10.3	7.59
pk. jerk	0.956	0.756	0.518	0.962
tot. jerk	70.6	43.1	8.35	127
tot. distance	229	207	254	222
tot. steering	183	67.5	84.5	181
runtime	449	346	362	471
onlane rate	0.225	0.991	0.943	0.992
col. rate	0.13	0.84	0.74	0.29

TABLE VI: DQN-M trained on *ComplexRoads* evaluated in other various environments

Indicator	Racetrack	Roundabout	Merge	Intersection
speed	10.2	8.3	30.6	10
tot. distance	180	200	377	59.3
runtime	275	349	185	89.5
onlane rate	0.998	0.602	0.935	0.998
col. rate	0.92	0.79	0.3	0.52

TABLE VII: TRPO-M trained on *ComplexRoads* evaluated in other various environments

Indicator	Racetrack	Roundabout	Merge	Intersection
speed	9.99	8.95	29.8	10.3
tot. distance	130	195	339	59.7
runtime	222	289	172	87.3
onlane rate	1*	0.647	0.996	0.999
col. rate	0.82	0.76	0.1	0.51

TABLE VIII: DQN-M trained on Roundabout evaluated in other various environments

Indicator	Racetrack	Merge	Intersection
speed	10.7	30.6	10.1
tot. distance	156	335	22.3
runtime	224	164	32.6
onlane rate	0.954	0.955	0.968
col. rate	0.97	0.2	0.05

TABLE IX: TRPO-M trained on Intersection evaluated in other various environments

Indicator	Racetrack	Merge	Roundabout
speed	9	30.9	8.91
tot. distance	137	477	236
runtime	253	228	345
onlane rate	0.999	0.97	0.527
col. rate	0.57	0.1	0.68

TABLE X: TRPO-M trained on Merge evaluated in other various environments

Indicator	Intersection	Racetrack	Roundabout
speed	9.87	9.85	9.38
tot. distance	14.2	437	349
runtime	22.2	632	486
onlane rate	0.886	0.399	0.159
col. rate	0.06	0.16	0.38

TABLE XI: TRPO-M trained on Racetrack evaluated in other various environments

Indicator	Intersection	Merge	Roundabout
speed	9.69	29.7	7.38
tot. distance	50.9	304	113
runtime	79.3	154	239
onlane rate	0.996	0.971	0.849
col. rate	0.67	0.6	0.76

While there might be various ways to express that one model outperforms another, it is important to prioritize safety as the main concern. Therefore, the measured values that this study considers the most important are the onlane rate and the collision rate, which reflect driving safety. Other values, such as speed or jerk, are less important but can be compared in cases where the onlane and collision rates are similar.

From Tables I, II, III, IV and V, one can see that in most cases the DQN with modified reward function (DQN-M) and the TRPO with modified reward function (TRPO-M) outperform the DQN and TRPO models with the baseline reward functions, especially with regards to the onlane rate.

Between the DQN and TRPO models, the models trained by TRPO tend to perform somewhat better in most cases.

Furthermore, looking at Tables VI, VII, VIII, IX, XI and X, it is observed that the models trained on *ComplexRoads* indeed tend to perform better than the other models in the cross-evaluation, especially in keeping a high onlane rate. This is due to various traffic situations represented in the *ComplexRoads* environment, as well as the fact that the starting location of the car during training on *ComplexRoads* was randomized, meaning that the car can experience various driving situations. This will also prevent the model from merely 'memorizing' the environment, but instead learning better to master the maneuvers to interact with the randomly generated environments.

Due to the size of *ComplexRoads*, training on it was very computationally intensive, especially with a large amount of simulated surrounding cars. Non-ego cars get destinations assigned randomly and drive around scripted, meaning they follow deterministic driving rules to drive 'perfectly' and receive a new destination upon reaching the previous one. Thus, this study opted to train the model with a relatively few surrounding cars, meaning that the model does not get to interact with other cars as often as in the other environments. Due to this, it resulted in a higher collision rate when evaluated in the other environments with more surrounding cars. When the computational resource is abundant, by adding more surrounding cars into the *ComplexRoads* environment, this reduced awareness of the ego car can be reduced.

All in all, it is verified that the designed *ComplexRoads* indeed contributes to the training of a more flexible and adaptive driving model. All the testing scenarios and results are better demonstrated in the appendix with the demo videos also provided at https://shorturl.at/oLP57.

V. CONCLUSION

This study first summarized the utilization of DRL in every specific automated driving task, e.g., lane-keeping, lane-changing, overtaking, and ramp merging, then customized and implemented two widely used DRLs, i.e., DQN and TRPO to tackle various driving maneuvers and carried out a comprehensive evaluation and comparison on the model performance. Based on `highway-env`, a modified and upgraded reward function was designed for training the DRL models. Furthermore, a new integrated training environment, *ComplexRoads*, was constructed, together with several built-in functions were upgraded. Through various experiments, it is verified that the models trained using the modified reward generally outperformed those with the original baseline reward and the newly constructed *ComplexRoads* demonstrated effective performance in training a uniform model that can tackle various driving tasks rather than one specific maneuver. As a preliminary study, the findings will provide meaningful and instructive insights for future studies towards developing automated driving with DRL and simulation.

One feature that was implemented by this study but was removed due to time constraints and lack of computational resources, was training the cars to reach a specific destination in the designed *ComplexRoads* environment, which requires more interactions of training and perhaps the implementation of a path finding and optimization algorithm. In particular, providing a metric distance from the ego car to the destination, incentivized the car to take road options which seem to directly reduce the distance, which means the car often choose poorly and got punished by the distance increasing before decreasing. While this approach might work for grid like city structures it confuses the learning process in general. Nevertheless alternative direct navigation reward designs are a very interesting direction for further research.

References

[1] C. Badue, R. Guidolini, R. V. Carneiro, *et al.*, "Self-driving cars: A survey," *Expert Systems with Applications*, vol. 165, p. 113 816, 2021.

[2] Q. Rao and J. Frtunikj, "Deep learning for self-driving cars: Chances and challenges," in *Proceedings of the 1st international workshop on software engineering for AI in autonomous systems*, 2018, pp. 35–38.

[3] D. Silver, A. Huang, C. J. Maddison, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[4] K. Shao, Z. Tang, Y. Zhu, N. Li, and D. Zhao, "A survey of deep reinforcement learning in video games," *arXiv preprint arXiv:1912.10944*, 2019.

[5] B. R. Kiran, I. Sobh, V. Talpaert, *et al.*, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 6, pp. 4909–4926, 2021.

[6] Z. Zhu and H. Zhao, "A survey of deep rl and il for autonomous driving policy learning," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 9, pp. 14 043–14 065, 2021.

[7] Y. H. Khalil and H. T. Mouftah, "Exploiting multimodal fusion for urban autonomous driving using latent deep reinforcement learning," *IEEE Transactions on Vehicular Technology*, 2022.

[8] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "End-to-end deep reinforcement learning for lane keeping assist," *arXiv preprint arXiv:1612.04340*, 2016.

[9] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *arXiv preprint arXiv:1704.02532*, 2017.

[10] P. Wang, C.-Y. Chan, and A. de La Fortelle, "A reinforcement learning based approach for automated lane change maneuvers," in *2018 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2018, pp. 1379–1384.

[11] J. Zhang, C. Chang, X. Zeng, and L. Li, "Multi-agent drl-based lane change with right-of-way collaboration awareness," *IEEE Transactions on Intelligent Transportation Systems*, 2022.

[12] M. Kaushik, V. Prasad, K. M. Krishna, and B. Ravindran, "Overtaking maneuvers in simulated highway driving using deep reinforcement learning," in *2018 ieee intelligent vehicles symposium (iv)*, IEEE, 2018, pp. 1885–1890.

[13] P. Wang and C.-Y. Chan, "Formulation of deep reinforcement learning architecture toward autonomous driving for on-ramp merge," in *2017 IEEE 20th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2017, pp. 1–6.

[14] D. Isele, R. Rahimi, A. Cosgun, K. Subramanian, and K. Fujimura, "Navigating occluded intersections with autonomous vehicles using deep reinforcement learning," in *2018 IEEE international conference on robotics and automation (ICRA)*, IEEE, 2018, pp. 2034–2039.

[15] Y. Guo and J. Ma, "Drl-tp3: A learning and control framework for signalized intersections with mixed connected automated traffic," *Transportation Research Part C: Emerging Technologies*, vol. 132, p. 103 416, 2021.

[16] Q. Hu and W. Yue, *Markov decision processes with their applications*. Springer Science & Business Media, 2007, vol. 14.

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

[18] A. Alizadeh, M. Moghadam, Y. Bicer, N. K. Ure, U. Yavas, and C. Kurtulus, "Automated lane change decision making using deep reinforcement learning in dynamic and uncertain highway environment," in *2019 IEEE intelligent transportation systems conference (ITSC)*, IEEE, 2019, pp. 1399–1404.

[19] Q. Liu, F. Dang, X. Wang, and X. Ren, "Autonomous highway merging in mixed traffic using reinforcement learning and motion predictive safety controller," in *2022 IEEE 25th International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2022, pp. 1063–1069.

[20] P. Polack, F. Altché, B. d'Andréa-Novel, and A. de La Fortelle, "The kinematic bicycle model: A consistent model for planning feasible trajectories for autonomous vehicles?" In *2017 IEEE intelligent vehicles symposium (IV)*, IEEE, 2017, pp. 812–818.

[21] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, "Trust region policy optimization," in *International conference on machine learning*, PMLR, 2015, pp. 1889–1897.

[22] J. Fan, Z. Wang, Y. Xie, and Z. Yang, "A theoretical analysis of deep q-learning," in *Learning for Dynamics and Control*, PMLR, 2020, pp. 486–489.

[23] D. H. P. C. C. (DHPC), *DelftBlue Supercomputer (Phase 1)*, https://www.tudelft.nl/dhpc/ark:/44463/DelftBluePhase1, 2022.