Promises and Challenges of Reinforcement Learning Applications in Motion Planning of Automated Vehicles

Nikodem Pankiewicz[1,2](✉), Tomasz Wrona[1,2], Wojciech Turlej[1,2], and Mateusz Orłowski[1,2]

[1] AGH University of Science and Technology, Krakow, Poland
[2] Aptiv, Krakow, Poland
{nikodem.pankiewicz,tomasz.wrona, wojciech.turlej, mateusz.orlowski}@aptiv.com

Abstract. As automated driving development progresses forward, novel methods are required to handle the vastness of possible road situations and to face end user's high demands. Trying to solve the problem of motion control involving decision making and trajectory planning it is reasonable to take into consideration reinforcement learning as a viable approach. In this paper, we present the promises reinforcement learning can bring to an automated driving domain and the list of challenges we encountered during our work. We address the issues related to the environment definition, sample efficiency, safety and explainability.

Keywords: Automated driving · Behavior planning · Reinforcement learning · Real-world

1 Introduction

Applying reinforcement learning (RL) methods to solve complex tasks is becoming increasingly popular and progressing research constantly delivers new state-of-the-art algorithms [15,23,30]. In some of these applications, RL-based solutions exceed even the human performance [10]. However, the vast majority of the considered problems were placed in the virtual domain, which usually takes the form of a computer game or a simulation, while the progress in solving problems in the real world domain is not that advanced. This immaturity is a concern for the industry, where the application potential is significant.

One of the potential areas of reinforcement learning methods application is behavior and trajectory planning of the automated vehicles. Due to the problem's complexity, rules-based deterministic algorithms may not be sufficient to achieve desirable system's performance. Therefore, a need for data-driven methods emerges and reinforcement learning seems to be a promising approach.

As a research group involved in the development of automated vehicle technology, we present our motivation and problems behind the application of reinforcement learning to this area. In this paper, we refer to the publication [13],

L. Rutkowski et al. (Eds.): ICAISC 2021, LNAI 12855, pp. 318–329, 2021.
https://doi.org/10.1007/978-3-030-87897-9_29

in which the authors outlined nine major problems of applying reinforcement learning in a real world setting. We describe the problems we encountered and investigated during the implementation of the behavior planning functionality in a close to production ready system.

1.1 Our Work

In this section we present an architecture of our system involving reinforcement learning agent in the place of behavior planning module of an automated driving stack.

Motion Planning System. Because of the reasons described in more detail in Sect. 3.4, we decided to create a hybrid system consisting both of reinforcement learning and model-based algorithms. The motion planning system is fed by a perception stack, which provides a representation of an ego car's surroundings, such as other road users and a road itself. The system consists of four main entities: the behavior planning module, the safety framework, the trajectory planning module and the control block. The architecture is presented in Fig. 1. In our case, the behavior planning module, providing a car with a high-level description of an intention, like a maneuver to execute or a speed recommendation, has a form of an reinforcement learning agent. The returned behavior is then realized by model-based methods responsible for trajectory planning and control, which are under the supervision of the safety framework. The safety framework itself, heavily based on the ideas presented in [31], along with an additional set of deterministic rules based on traffic law, assures safety of the AI-based behavior planning as well as the trajectory planning. The behavior planning module can also be additionally supplied with signals from the mission planning module, responsible for producing sub-goals necessary to follow the defined path.

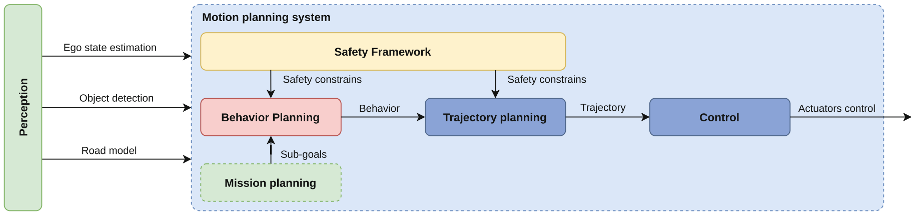

Fig. 1. The motion planning architecture: the RL-based behavior planning module (in red) and the model-based trajectory planning and control modules (in blue). (Color figure online)

Reinforcement Learning Environment. To train a reinforcement learning agent responsible for behavior planning we design a custom environment based on a proprietary closed-loop simulation tool [6]. The environment consists of multiple blocks, which we listed below.

- **Behavior decoding** decodes an action (outputted from a policy neural network) to a semantic value, like a maneuver to execute or a velocity set point (see Sect. 3.3).
- **Trajectory planning** establishes a specific realization of a behavior and returns a continuous trajectory to be followed by the agent.
- **Control** produces low-level control signals (e.g. throttle, braking, steering wheel angle) to keep the car on its reference trajectory.
- **Simulation** delivers a closed-loop simulation of vehicle dynamics and road users interactions.
- **Perception simulation** disturbs the perfect ground truth obtained from the simulation using high-level sensor models (see Sect. 3.5).
- **Observation encoding** encodes the state of the environment to a form easily interpretable by the policy network. The design of this transformation is crucial for the policy to be able to generalize well in various situations (see Sect. 3.3).
- **Reward calculation** defines a numerical reward signal for the agent, designed to promote the agent for keeping a cruising velocity and reaching predefined lane-based goals, while maximizing comfort and safety of passengers (see Sect. 3.6).

By defining the environment in such way, we are able to control the current challenge for the agent as well as experiment with different versions of each module.

Training Setup. Having the environment defined, we utilize standard reinforcement learning algorithms, such as DQN [23], PPO [30] or SAC [15] to train the behavior planning agent. For better generalization and more efficient training, these algorithms were scaled up to support a distributed setup. Our best performing setup is a combination of PPO algorithm and multiple state-of-the-art methods available in the literature. In [28] we presented our previous results with DQN algorithm, along with the more detailed description of the environment definition, including the observation and action spaces and the reward shaping mechanism (Fig. 2).

2 Promises

Using data-driven methods over other methods in behavior and trajectory planning modules brings several promises. The most obvious one is the transfer of the responsibility for the design of driving policy rules and actions from the group of human experts to the computer program. This design might be done based on a large amount of data collected with the help of a realistic virtual simulation. A good example might be Waymo, a self-driving company, which states that their systems drive 20 million miles in a simulation each day [4]. Still, the choice between supervised learning and reinforcement learning methods is an open question.

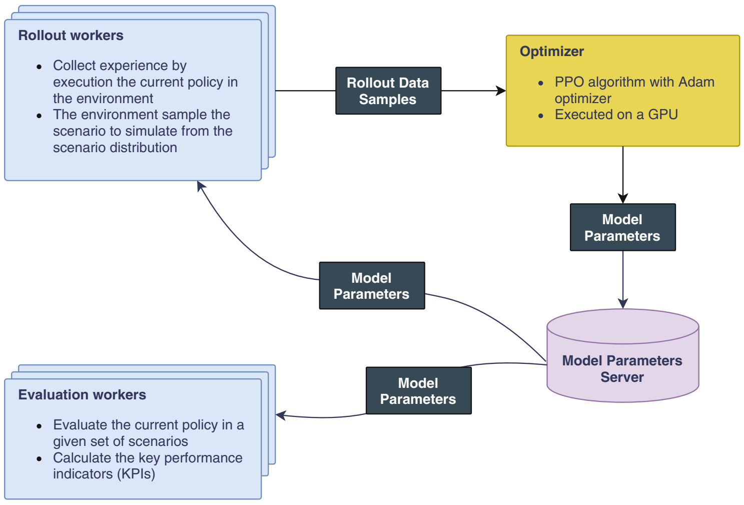
Rollout workers
Collect experience by execution the current policy in the environment
The environment sample the scenario to simulate from the scenario distribution
Rollout Data Samples
Optimizer
PPO algorithm with Adam optimizer
Executed on a GPU
Model Parameters
Model Parameters
Model Parameters Server
Model Parameters
Evaluation workers
Evaluate the current policy in a given set of scenarios
Calculate the key performance indicators (KPIs)

Fig. 2. The architecture of the behavior planning agent training setup.

Supervised learning methods, such as e.g. behavior cloning [33], might be more interesting due to their higher stability of the training process, but they require human interaction in the data collection process, thus making this process significantly slower and more expensive [1]. Furthermore, when the trained policy is found to be unsatisfactory, the data has to be collected again. In the case of reinforcement learning, for some algorithms, the data can be reused by recalculating the rewards only.

The more important difference, however, is with the trained policy itself. With supervised learning, we are limited to a single policy demonstrated by a driver. Moreover, it is unlikely that the driver's policy is the optimal one. This is due to the fact that it is usually hard to decide, which action is the best decision at a given moment, and how this action will affect the future states. This problem is known as the credit assignment problem, and many reinforcement learning algorithms tackle this problem by default. Yet another problem we observed is the difference between the perception system of a human and an automated vehicle. The drivers' decisions are affected by human factors such as lack of attention, cognitive distraction [21,36] or aggressiveness level [22]. That and other not listed discrepancies in perception causes the distributional shift between learning and evaluation domain of automated vehicle. Finally, an exploration of corner cases (accidents, dangerous or unusual situations), even when carried out in the virtual simulation, is more efficient with reinforcement learning methods, as humans are willing to risk less, than computer programs.

3 Challenges

3.1 Training Off-Line from the Fixed Logs of an External Behavior Policy

Fixed logs from driving are valuable assets in the entire learning process. Such logs can be used in a supervised manner to train an initial policy with behavior cloning [33], as a part of imitation learning process [34,35] or as an additional exploration guiding for policy optimization [20].

A good example of successful logs utilization is the work [11] where the team from Waymo used 30 million samples of car's trajectories to train a neural network policy, which was responsible for generating a future trajectory for a vehicle. As the authors admitted, the model was good enough to drive a vehicle successfully, however it was not fully competitive with the deterministic motion planning despite using a considerable amount of data.

Besides the amount of data, the samples should be also of sufficient quality. Erroneous actions resulting from driver's mistakes are misleading during training. Additionally, data should include states in which a reasonable driver will never be situated. These states provide, however, valuable inputs to the learning process as they alleviate distributional shift between the learning and the operating domains.

3.2 Learning on the Real System from Limited Samples

The process of developing functionalities, capable of operating in the real world, based solely on simulation is burdened with the simulation to reality (sim-to-real) gap issue. Therefore, a logical step to close this gap is to fine-tune the policy on public roads. In this process, a most recent driving policy is used to interact with the target environment, explore it and collect data for the next training iteration. The relatively slow rate of data collection, thus the limited number of samples, and willingness of data re-usage suggest utilization of off-policy methods, such as DQN [23] or SAC [15], which latter's efficiency in training the physical robots was demonstrated in [16].

The complication which arises in such an approach is a risk resulting from exploiting a deficient policy in a vulnerable environment. The application of such a solution requires ensuring the safety of the environment influenced by the agent. In our work, this approach is viable because the agent's action is secured by a deterministic trajectory planning and its safety module (discussed in Sect. 3.4).

In case of recognizing performance drops in some situations during evaluation, the process enables counteracting them by involving more vehicles in those problematic scenarios, which can be later used to improve the policy in those cases.

3.3 High-Dimensional Continuous State and Action Spaces

While designing state spaces for robotic applications it is reasonable to convert raw sensor inputs to another, meaningful representation, e.g. camera images to a set of detected objects or a lidar points cloud to a spatial map of surroundings. That way we decrease the amount of black-box models in the motion planning module, which significantly increases its explainability. Moreover, the specific representation of the input data contributes to a better generalization of a neural network and speeds up its training. We use this approach in our work, where we represent the environment with the high-level properties of the ego vehicle (e.g. orientation, velocity), the target vehicles, and the road model.

A reasonable action space for an automated vehicle motion planning module must consist of at least one control signal for each of the two spatial dimensions: longitudinal (velocity, acceleration, throttle, braking) and lateral (heading, steering angle). Another approach is to determine a set of waypoints, which describe consecutive features of a trajectory: velocity and acceleration values sampled along a reference path.

As mentioned in Sect. 1.1, we split the control decision component into two modules: behavior planning and trajectory planning. The behavior planning module is further combined with strategic and tactical planners. The former outputs one maneuver (e.g. follow lane, change lane left, prepare to change lane left) from the list of the currently available maneuvers. The state of the list depends on the state of the current situation on the road and is strictly controlled by the safety framework. The latter produces a combination of additional guiding signals for the trajectory planning module, such as a velocity set point or a lane bias. The real values are discretized with a resolution found empirically, having in mind the trade-off between the model complexity and the maneuvers' flexibility.

3.4 Satisfying Safety Constraints

All components of a modern vehicle must meet strict safety requirements. Both hardware and software are subject to validation accordingly to standards such as ISO 26262 (Functional Safety) or ISO/PAS 21448 (Safety of the Intended Functionality) and therefore are heavily tested before they reach an end customer. With the advent of advanced systems, which take more and more responsibility for control of a vehicle, safety considerations for highly automated vehicles is a complex topic in itself.

Looking for the required performance with regards to safety it is reasonable to refer to the human drivers. While car accidents are caused to over 1.3 million deaths annually [27], a fatal accident rate of a single vehicle is relatively low, reaching on average 1 fatality per 100 million miles in US [8]. Since statistics on the road-related accidents are shaping expectations regarding maximum failure rates of automated vehicles, a rarity of such events poses a great challenge to companies developing such systems. Showing with reasonable confidence that a system has a significantly lower failure rate in terms of fatal accidents through end-to-end test drives would require billions of miles of test drives [19].

A relative rarity and importance of severe accidents create a challenge not only for testing and validation but also for the development of such systems. Learning an end-to-end RL-based planning system to drive with reasonably low severe collision rate requires modeling of reward function with trade-off between efficiency and safety or using constrained RL [7]. However, to liberate the RL part from the problem which could be formulated in a deterministic way, we decided that the reinforcement learning system should not be formally responsible for the safety aspects of driving.

Another observation may be made with respect to traffic regulations. Training an agent to be compliant with them with the use of a reward signal will always end up in their approximations, which is unacceptable. Fortunately, it is straightforward to design a model-based system obeying such rules by definition.

The points above suggest that end-to-end planning systems based on reinforcement learning present several serious issues. It seems that the desired system should be rather a hybrid approach, composed of model-based and machine learning modules. The interface between the reinforcement learning module and the deterministic part of the system has to satisfy two contradicting objectives. On the one hand, the interface has to be constrained enough to not put safety requirements onto the reinforcement learning part. On the other, it has to be open enough to benefit from data-driven methods in planning and assure a sufficient level of policy transparency, allowing efficient and explainable learning.

An interesting approach intended for providing safety guarantees in path planning systems with reinforcement learning elements was proposed in [31]. Authors formulate a set of safety rules that are used to derive deterministic constraints for a path planning algorithm. The rules describe a safety envelope, i.e. a constrained area of a state space, which guarantees the existence of collision-free trajectories in all reasonable situations. The reasonable situations are the ones in which road users follow traffic laws and formalized common-sense traffic rules. Early detection of safety envelope violations, caused by a controlled agent, allows to execute predefined emergency responses in time and avoid collisions. Similar concepts were proposed also in [26] and [29].

The described approach can fulfill two major roles in the RL-based path planning system: limiting the available action space and triggering execution of emergency trajectories in unsafe situations (that may be caused by both ego's actions and other agents' behaviors). The safety envelope provides transparent, testable safety constraints that protect an agent against performing unsafe maneuvers and enforces proper responses when the constraints are violated.

This approach alone provides a certain level of protection against collisions, but still, overall system performance in terms of avoiding unsafe situations can be improved in a training process. Since emergency actions typically consist of severe braking, which is sub-optimal in terms of passengers' comfort and driving efficiency, a properly trained agent will proactively avoid potentially unsafe actions. An example of such behavior may be to switch a lane away from a merge-in lane to avoid potentially unsafe situations involving slow vehicles merging into highway traffic.

Proactive safety behaviors emerging in an RL-based system may have particular significance in a context of a realistic sensor stack with limited performance. Simulating typical sensors' error patterns and occlusions in a training setup may significantly decrease the impact of perception issues on overall system safety. An agent trained in such environments may learn to avoid situations that are correlated with hazardous perception errors, as well as to perform human-like visibility-enhancement actions, such as biasing on its lane to unveil occluded areas of the environment.

3.5 Partial Observability and Non-stationarity

In the automated driving domain, both partial observability and non-stationarity are major issues significantly affecting the trained policy's performance. They are a result of different factors, from the physical world barriers to software and hardware limitations. In this section, we summarize what type of problems we observed during our research, what solutions we applied, or what solutions we propose.

Occluded Objects. The first problem is occluded objects: other vehicles, pedestrians, traffic signs and lights, lane markings, and other elements of infrastructure. One possible general solution might be to involve V2V or V2X communication [18] to exchange the information between the objects, even when they are not visible to each other. Another option is to assume worst-case scenarios, as proposed in [31] and discussed in Sect. 3.4, and act appropriately, but this might lead to overly protective policies. We implemented this by injecting worst-case object hypotheses on occlusion edges into perception results and allow agents to learn avoiding potentially unsafe regions to minimize effects on driver's comfort or total efficiency.

Perception Issues. The second problem is the imperfection of a sensing stack. A typically automated vehicle prototype is nowadays equipped with a combination of multiple cameras, radars, and lidars. Each type of the sensors has its advantages and disadvantages, thus the common approach is to fuse data from multiple sources to obtain a high fidelity representation of the vehicle's surroundings. In practice, false positives (ghost objects), false negatives (missing objects), or significant measurement uncertainties are still present in the fused representation. To address these problems in simulation, sensor models can be introduced to imitate the behavior of the real-world sensing suite. A lot of work has already been done in the area of detailed sensor simulation [14, 24]. However, due to the detailed simulation, the proposed solutions are usually computationally expensive, what significantly extends the simulation time, thus slow down the learning process. To alleviate this issue, we propose the usage of simple, high-level sensor models, which skip understanding the underlying physical principles of a given sensor (e.g. how radar wave is reflected from an object and how it is later tracked), and instead utilize statistical models for each common problem (ghost objects, uncertainty, etc.; Fig. 3).

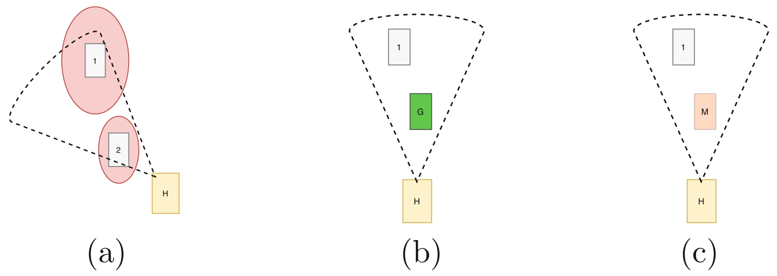

Fig. 3. The high-level sensor models: a) a measurement uncertainty (in red) is being applied to target objects (in gray); b) a ghost object (in green) is being inserted into the sensor's FOV; c) a missing object (in red) is being removed from the sensor's FOV. (Color figure online)

Intentions. The third problem is missing information about other road users' intentions. For the same observation of a state, the consecutive states might differ between episodes, depending on the *hidden state* of other drivers. One solution might be to add a pipeline responsible for the behavior or trajectory prediction of road users. This introduces an extra computing overhead and is a challenging task in itself, but significantly increases the model's explainability. Another solution is to let a policy's network spot the interesting connections on its own and store them in memory blocks.

In the previous paragraphs, we mentioned memory blocks as a solution for partial observability and non-stationarity of the environment. The obvious solution is to use modules such as LSTM [17] cells, but they lack interpretability. Frame stacking or putting historical data into the observation seems to be more suitable for safety-critical systems, however, in order to achieve satisfying policies, they might need to generate their own signals (e.g. information about a long-term decision taken in the past) and pass them between consecutive timesteps. Ideally, to achieve high interpretability, these signals have to be identified and defined by a human in advance. This, however, limits the capabilities being possible to be trained by the policy.

3.6 Unspecified and Multi-objective Reward Functions

Designing a reward function is another challenge that determines the success of the implemented solution. Drivers while making their decisions, take into account many aspects, which should be identified and represented during the reward function design.

Therefore, the reward function should be a compromise between pro-active safety assurance, performance on reaching destination targets, efficiency (time, fuel), comfort, and a general impact on a traffic flow. However, reward hacking [9] is a known problem in reinforcement learning, which causes an agent to not behave as expected, even though it achieves high rewards. We observed this in our work multiple times, e.g. when the agent was performing too frequent changes of its current maneuver. It is important to remember that the reward function designer must adequately prioritize and balance the individual parts of the reward signal or use meta-learning techniques.

3.7 Explainability

Explainability is a crucial requirement when working with safety-critical systems such as automated vehicles. In case of a system failure, whether in a form of an accident or a dangerous situation, a testing team must have a tool to understand why the situation happened, which decision was wrong, and what signals attributed most to this decision. Fortunately, the topic is being extensively researched recently and a good survey on available methods was presented in [25], as well as open-source tools [2,5] were released.

In this paper, we focus on neural network-based policies, where the input signals are high-level signals outputted from other systems (perception, localization). This is in contrast to end-to-end methods (e.g. [12]), where the input data is usually raw data and its interpretation is rather an uneasy task. In our situation, the input signals are labeled, and we can use simple methods such as e.g. Integrated Gradients [32] to calculate each input neuron's attributation to the output signal and provide meaningful insight on the policy's motivation.

With this methodology, we can navigate through a complete episode and investigate, what caused a problematic decision at any given time step, and how the decision would change, if we change the observation. We found this approach successful for some tasks and open-sourced our initial version of the tool [3]. We, however, encountered the problem with understanding neurons' state in LSTM cells. As mentioned in Sect. 3.5, there is a need for more explainable memory blocks.

4 Summary

In this paper, we presented the promises we see standing behind the application of the reinforcement learning method in motion planning. We also listed the challenges we faced during our research on the behavior planning module and described our solutions for these issues. Based on our observations, we believe that the current state of the research allows us to successfully deploy reinforcement learning in real-world automated driving. However, some work has to be done yet to e.g., ensure better explainability of an artificial agent and to provide an easier comparison between different versions and methods of the trained agent. As a consequence of the unpredictability of RL agents, we presented our method of satisfying safety constraints, but this approach still requires exhaustive testing in an operating domain. Additionally, the formulation of reward function remains an open research topic. As an active research group, we will continue our research towards more challenging traffic scenarios such as occurring on urban roads. Our future research will also be devoted to the explainability problem of RL agents.

References

1. Autonomous Vehicle Data Annotation Market Analysis. https://www.researchandmarkets.com/reports/4985697/autonomous-vehicle-data-annotation-market-analysis
2. Captum. Model Interpretability for PyTorch. https://captum.ai/
3. GitHub - iamhatesz/rld: A development tool for evaluation and interpretability of reinforcement learning agents. https://github.com/iamhatesz/rld
4. Off road, but not offline: How simulation helps advance our Waymo Driver. https://blog.waymo.com/2020/04/off-road-but-not-offline-simulation27.html
5. sicara/tf-explain: Interpretability Methods for tf.keras models with Tensorflow 2.x. https://github.com/sicara/tf-explain
6. Traffic AI – Simteract. http://simteract.com/traffic-ai/
7. Achiam, J., Held, D., Tamar, A., Abbeel, P.: Constrained policy optimization. CoRR abs/1705.10528 (2017). http://arxiv.org/abs/1705.10528
8. Administration, F.H.: Highway statistics, 2018. Technical report, Washington, DC: US Department of Transportation (2019)
9. Amodei, D., Olah, C., Steinhardt, J., Christiano, P.F., Schulman, J., Mané, D.: Concrete Problems in AI Safety. CoRR abs/1606.06565 (2016). http://arxiv.org/abs/1606.06565
10. Badia, A.P., et al.: Agent57: Outperforming the Atari Human Benchmark (2020)
11. Bansal, M., Krizhevsky, A., Ogale, A.S.: ChauffeurNet: Learning to Drive by Imitating the Best and Synthesizing the Worst. CoRR abs/1812.03079 (2018). http://arxiv.org/abs/1812.03079
12. Bojarski, M., et al.: End to End Learning for Self-Driving Cars (2016). http://arxiv.org/abs/1604.07316
13. Dulac-Arnold, G., Mankowitz, D.J., Hester, T.: Challenges of Real-World Reinforcement Learning. CoRR abs/1904.12901 (2019). http://arxiv.org/abs/1904.12901
14. Dworak, D., Ciepiela, F., Derbisz, J., Izzat, I., Komorkiewicz, M., Wojcik, M.: Performance of LiDAR object detection deep learning architectures based on artificially generated point cloud data from CARLA simulator. In: 2019 24th International Conference on Methods and Models in Automation and Robotics, MMAR 2019, pp. 600–605. Institute of Electrical and Electronics Engineers Inc. (2019). https://doi.org/10.1109/MMAR.2019.8864642
15. Haarnoja, T., Zhou, A., Abbeel, P., Levine, S.: Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor (2018). http://arxiv.org/abs/1801.01290
16. Haarnoja, T., et al.: Soft Actor-Critic Algorithms and Applications. CoRR abs/1812.05905 (2018). http://arxiv.org/abs/1812.05905
17. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997). https://doi.org/10.1162/neco.1997.9.8.1735
18. Jung, C., Lee, D., Lee, S., Shim, D.H.: V2x-communication-aided autonomous driving: system design and experimental validation. Sensors (Switzerland) **20**(10), 2903 (2020). https://doi.org/10.3390/s20102903, http://pmc/articles/ PMC7287954/?report=abstract www.ncbi.nlm.nih.gov/pmc/articles/PMC7287954/
19. Kalra, N., Paddock, S.M.: Driving to safety: how many miles of driving would it take to demonstrate autonomous vehicle reliability? Trans. Res. Part A Policy Pract. **94**, 182–193 (2016)

20. Kang, B., Jie, Z., Feng, J.: Policy optimization with demonstrations. In: 35th International Conference on Machine Learning, ICML 2018, vol. 6, pp. 3855–3869 (2018)
21. Kass, S.J., Cole, K.S., Stanny, C.J.: Effects of distraction and experience on situation awareness and simulated driving. Transp. Res. Part F: Traffic Psychol. Behav. **10**(4), 321–329 (2007). https://doi.org/10.1016/j.trf.2006.12.002
22. Lajunen, T., Parker, D.: Are aggressive people aggressive drivers? A study of the relationship between self-reported general aggressiveness, driver anger and aggressive driving. Accid. Anal. Prev. **33**(2), 243–255 (2001). https://doi.org/10.1016/S0001-4575(00)00039-7, https://linkinghub.elsevier.com/retrieve/pii/S0001457500000397
23. Mnih, V., et al.: Playing Atari with Deep Reinforcement Learning (2013). http://arxiv.org/abs/1312.5602
24. Molenaar, R., Van Bilsen, A., Van Der Made, R., De Vries, R.: Full spectrum camera simulation for reliable virtual development and validation of ADAS and automated driving applications. In: IEEE Intelligent Vehicles Symposium, Proceedings, vol. 2015-August, pp. 47–52. Institute of Electrical and Electronics Engineers Inc. (2015). https://doi.org/10.1109/IVS.2015.7225661
25. Molnar, C.: Interpretable Machine Learning (2019)
26. Nistér, D., Lee, H.L., Ng, J., Wang, Y.: The Safety Force Field. Technical report
27. World Health Organization et al.: Global status report on road safety 2018: Summary. World Health Organization, Technical report (2018)
28. Orłowski, M., Wrona, T., Pankiewicz, N., Turlej, W.: Safe and goal-based highway maneuver planning with reinforcement learning. In: Advances in Intelligent Systems and Computing, vol. 1196 AISC, pp. 1261–1274. Springer (2020). https://doi.org/10.1007/978-3-030-50936-1_105, https://link.springer.com/chapter/10.1007/978-3-030-50936-1_105
29. Pek, C., Althoff, M.: Computationally efficient fail-safe trajectory planning for self-driving vehicles using convex optimization. In: 2018 21st International Conference on Intelligent Transportation Systems (ITSC), pp. 1447–1454. IEEE (2018)
30. Schulman, J., Wolski, F., Dhariwal, P., Radford, A., Klimov, O.: Proximal Policy Optimization Algorithms (2017). http://arxiv.org/abs/1707.06347
31. Shalev-Shwartz, S., Shammah, S., Shashua, A.: On a Formal Model of Safe and Scalable Self-driving Cars (2017). http://arxiv.org/abs/1708.06374
32. Sundararajan, M., Taly, A., Yan, Q.: Axiomatic Attribution for Deep Networks. Technical report (2017)
33. Torabi, F., Warnell, G., Stone, P.: Behavioral cloning from observation. In: IJCAI International Joint Conference on Artificial Intelligence 2018-July(July), pp. 4950–4957 (2018). https://doi.org/10.24963/ijcai.2018/687
34. Via, E.: What would Π do?: Imitation Learning via off-policy Reinforcement Learning, pp. 1–13 (2019)
35. Wu, Y.H., Charoenphakdee, N., Bao, H., Tangkaratt, V., Sugiyama, M.: Imitation Learning from Imperfect Demonstration. Technical report. https://www.basketball-reference.com/leagues/NBA_stats.html
36. YoungPaul, K.L., Salmon, M.: Examining the relationship between driver distraction and driving errors: a discussion of theory, studies and methods. Safe. Sci. **50**(2), pp. 165–174 (2012)