On the Challenges of Safe and Scalable Reinforcement Learning for Automated Driving at Intersections

Danial Kamran, Marvin Busch, Tizian Engelgeh
Institute of Measurement and Control Systems,
Karlsruhe Institute of Technology (KIT), Karlsruhe, Germany
Email: danial.kamran@kit.edu, {marvin.busch, tizian.engelgeh}@student.kit.edu

***Abstract*—In this paper we address safety and scalability as the main challenges that are still existing for decision making policies based on reinforcement learning (RL) to be applied for automated driving. We show how Deep-sets structure which was used before for automated lane changes can also solve dynamic order and input variations for the RL agents used at occluded intersections where the number of lanes and vehicles can be different at different scenarios. According to our evaluation results, deep-sets DQN can learn an optimal policy faster, and becomes more stable comparing with the normal DQN agent.**

I. Introduction

A. Motivation

One of the most important challenges for automated driving is providing a safe and scalable policy which can handle complex situations with low or high traffic density and also different environments. In reality, there are several types of uncertainties that need to be considered for the decision making policy in order to generate safe and robust actions. These uncertainties can come from partial observation of the automated vehicle mainly because of sensor occlusions or unknown drivers intention. Reinforcement Learning is a suitable framework for learning optimal decisions for complex robotics tasks including automated vehicles [1, 2, 3, 4, 5, 6, 7, 8]. This framework helps to learn long term optimal decisions for different scenarios in automated driving such as yielding in an occluded intersection [1, 2, 3, 4, 8] or lane changes in highway [5, 6, 7] .

In this work we focus on automated driving at un-signalized intersections where the ego vehicle needs to yield to vehicles which are close to the intersection and leave when they are far enough. We try to address some main challenges that prevent reinforcement learning based policies to be applied in real world for such scenario. These challenges can be summarized as:

- Dynamic inputs: The number of cars, intersecting lanes and also their order inside the state representation can be different from the training data which can result in catastrophic decisions. Therefore, the learned policy should be scalable and robust to these changes.
- Partial observation in the environment: Because of occlusions some vehicles are not visible for the decision making policy. Also the intention of drivers (like being cooperative or aggressive) is unknown. Therefore, the learned policy should be able to handle such uncertainties in its inputs and provide safe but not conservative decisions.

B. Related Works

Figure 1 depicts the whole scenario and parameters we use in order to represent the state for our decision making agent as presented before in [8]. The main advantage of such state representation is that it can model different situations that can happen for an automated driving car at an occluded intersection. However, in contrast to grid based representation which assume a fixed 2d array for modeling the whole intersection as proposed in [1], the length of such representation depends on the maximum number of elements that can exist in the scenario and can be very big for dense traffics at complex intersections where a huge number of vehicles are driving at an intersection with three or four intersecting lanes. Such huge state vector can result in a big DQN network in order to process the list of all cars and can cause overfitting problems. Another problem that arises when using the standard architecture is the permutation of the input elements. In complex situations it is often not possible to form a uniform order of the elements within the input vector. Even if the problem is independent of the permutation of the elements, this will still lead to a higher learning complexity.

The deep sets architecture was developed to solve the problem of dynamic inputs for neural networks for general machine learning tasks in [9]. In [7] the idea of deep sets was used in the context of reinforcement learning for automated driving, where the agent had to learn optimal lane change decisions in highway scenario.

C. Contribution

In this paper, we propose a new architecture for the intersection scenario where the reinforcement learning agent can process data for multiple vehicles and also different intersection lanes using deep sets structure suited for the state modeling shown in Figure 1. We show how the deep-sets architecture fits very well to such state representation which categorizes input data into vehicles, lanes and ego vehicle information. Being permutation invariant and independent from the size of input

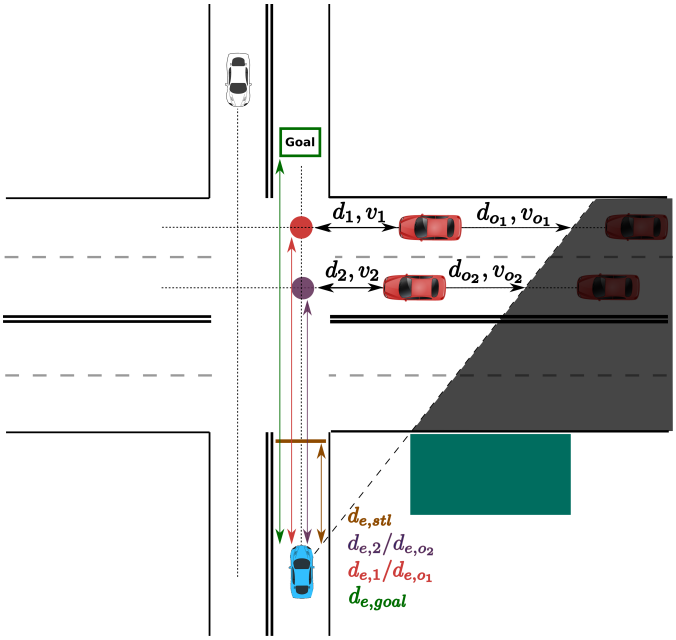

Figure 1: Overview of the yielding scenario we consider in our work and parameters used for state representation. Blue car is ego vehicle and red cars are relevant vehicles. Some cars are occluded and not visible to the ego vehicle. White car which has no potential conflict zone with the ego lane is discarded.

data, deep-sets architecture can help to learn a scalable and generic policy which is robust to the input order and stable at different traffic densities.

Moreover, we introduce yielding policies based on distributional reinforcement learning which can learn return distributions for each action instead of its expected value that can help to provide more robust policy against environment uncertainties.

II. Proposed Approach

A. State Representation

Using the model shown in Figure 1 for a scene at an occluded intersection with arbitrary number of intersecting lanes and vehicles, the state used to represent that scene is defined as below:

$$s_t = \begin{bmatrix} \overbrace{d_{\text{e,stl}}}^{ego} & \overbrace{d_1 \ \dots \ d_n}^{vehicles} & \overbrace{d_{o_1} \ \dots \ d_{o_m}}^{lanes} \\ v_{\text{e}} & v_1 \ \dots \ v_n & v_{o_1} \ \dots \ v_{o_m} \\ d_{\text{e,goal}} & d_{\text{e},1} \ \dots \ d_{\text{e},n} & d_{\text{e},o_1} \ \dots \ d_{\text{e},o_m} \end{bmatrix}^T$$

Such state representation has been previously used in [8] for a list based DQN architecture which assumes fixed number of input elements for cars and intersecting lanes. In that work, in maximum 5 vehicles and 4 lanes could be provided for the network and in case of having more vehicles, a criticality function will specify which vehicles are more important and should be represented for the DQN. However, in our deep-sets based DQN model, we can increase the maximum number of input elements to a big number (maximum 16 vehicles and 4 lanes) because the input space dedicated for the DQN model does not depend on the size of input elements anymore.

B. Deep-set based State Processing

In a deep-sets based state processing architecture, the state $s_t = (x_{static}, X_{dyn})$ is divided into a static and a dynamic part. The static part x_{stat} corresponds to the values of the state which are used in the same way in every situation. In our intersection scenario the static part of the state refers to the information about the ego vehicle such as its distance to the stop-line, distance to the goal and its velocity. This information is processed by ϕ_{ego} network and then concatenated with processed information from dynamic inputs. The dynamic part $X_{dyn} = [x_1, x_2, ..., x_n]$ consists of the vectors of the n objects and can vary in size as well as in the permutation of the objects. In the intersection scenario, dynamic input refers to the information about vehicles and also intersecting lanes which can be different at each situation. The deep sets architecture only applies to the dynamic part which consists of the neural networks ϕ and ρ and a permutation-invariant operator. We dedicate two deep-sets architectures for the two dynamic input categories at intersection scenario $X_{dyn} = (X_{veh}, X_{lane})$ which are vehicles' input X_{veh} and intersecting lanes X_{lane}. Therefore, all vehicles and lanes data are processed as follows:

$$\psi_{veh}(X_{veh}) = \rho_{veh}\Big(\sum_{veh_i \in X_{veh}} \phi_{veh}(veh_i), \phi_{ego}(x_{static})\Big) \tag{1}$$

$$\psi_{lane}(X_{lane}) = \rho_{lane}\Big(\sum_{lane_i \in X_{lane}} \phi_{lane}(lane_i), \phi_{ego}(x_{static})\Big) \tag{2}$$

Figure 2 shows the overall structure of the proposed deep-sets architecture in order to process static and dynamic parts of the input state. In contrast to [7] which only provide static input for the last layer connected to the DQN network, we provide processed information of static input ($\phi_{ego}(x_{static})$) for each input category processing network (ρ_{veh} and ρ_{lane}) which help to process dynamic input data relative to the ego state at the intersection. In our implementation we used sum of vectors as the permutation invariant operator, but any other permutation invariant such as polling could also be used.

C. Reward Function

The reward function we used in our DQN is designed in order to punish collisions and motivate fast but safe driving through the intersection. For that purpose, we use this reward function:

$$r(t) = \begin{cases} -1 & \text{on collision,} \\ 1 & \text{on success,} \\ 0 & \text{on non-terminating steps} \end{cases} \tag{3}$$

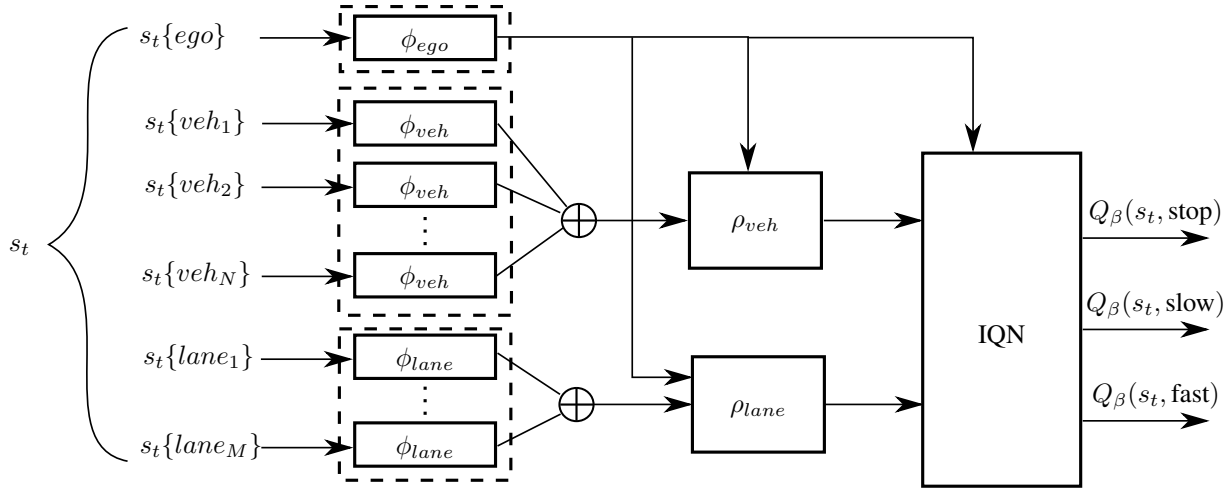

Figure 2: Proposed Deep-sets architecture with the neural networks ϕ_{veh} and ρ_{veh} for processing vehicles' data and ϕ_{lane}, ρ_{lane} for processing information of intersecting lanes. The permutation invariant operator is shown as $\bigoplus$.

D. Action Space

As proposed in [8], for automated driving at occluded intersections we can learn a policy generating high level actions instead of vehicle acceleration control commands. For that, three actions as high level decisions are defined for our RL algorithm:

- Stop: By this action, the ego vehicle should reach zero velocity as fast as possible. It can interpreted as a give-way high level action at the intersection where ago vehicle should yield to other vehicles which are close to the intersection.
- Drive slow: The ego vehicle should reach a fixed slow velocity (1 m/s) meaning that the situation is still unclear and it should drive slowly to gather more information.
- Drive fast: The ego vehicle should reach a fixed high velocity (5 m/s) meaning that there is no vehicle or it can get way from other vehicles which are far from the intersection.

E. Implicit Quantile Networks (IQN)

We model the scenario of automated driving at intersection as a Markov Decision Process (S, A, R, P, γ) where S and A are state and action spaces and R is the reward function as discussed in previous sections. P is the transition function as $Pr(.|s_t, a_t)$ and γ is the discount factor and is a value between 0 and 1. Assuming agent is following a policy π, the future return for each action a_t at each state s_t is defined as $Z^\pi(s_t, a_t)$ and the expected value of this random variable is defined as the value function of the policy:

$$Q^\pi(s_t, a_t) := \mathbb{E}Z^\pi(s_t, a_t) = \mathbb{E}[\Sigma_{i=t}^{T} \gamma^{(i-t)} r(s_i, a_i)], \quad s_t \sim P(.|s_{t-1}, a_{t-1}), a_t \sim \pi(.|s_t) \tag{4}$$

One of the main difficulties with the DQN approach is tuning the reward function in order to prevent risky behaviors and motivate those safe actions that also provide higher utility. The policy learned this way would definitely perform more stable and safer in reality where several uncertainties can occur even when they are slightly different from the training environment. However, due to the fact that DQN tries to maximize expected future return and neglects return distributions, there are always some situations where safety is sacrificed in order to have higher utility. In order to solve this problem and provide risk aware policies that are able to distinguish between risky actions with higher average utility and safe actions which may have lower average utility, we use distributional reinforcement learning. Therefore, instead of learning expected value for future returns of each action as $Q(s_t, a) = \mathbb{E}[Z(s_t, a)]$, the agent learns return distributions $(Z(s_t, a_t))$. Such implementation helps to learn a more robust policy which is less sensitive to hyperparameter changes.

Among different variations of distributional reinforcement learning that have been proposed like categorical distribution for fixed set of equidistant points (C51) [10], quantile regression (QR_DQN) [11], we use Implicit Quantile Network (IQN) [12] in order to estimate return distributions. This approach does not require fixed output distribution range since it learns the quantile function $F_Z^{-1}(\tau)$ for the return as the random variable Z where τ is a uniform sample $\tau \sim U([0, 1])$.

III. Evaluations

A. Simulation Environment

For simulating the intersection scenario with occlusions we used Carla simulator [13] and selected one of un-signalized intersections where the ego vehicle should drive at one intersection with up to four intersecting lanes with different number of vehicles. Training phase consists of more than 5000 episodes. At the beginning of each training episode, ego vehicle and random number of other vehicles are positions at random distances from the intersection. Each vehicle has random desired speed and is randomly assigned to drive on one of intersection lanes. Also for each episode, a virtual obstacle

Figure 3: Top view images from the simulation used for training. Images are from one episode where ego vehicle (red vehicle) stops behind stop line in order to yield to other vehicles (image left). In the middle image it starts driving through the intersection and reaches the goal point (right image). Some vehicles are not visible for the reinforcement learning agent because they are occluded by the virtual obstacle which is randomly generated for each episode.

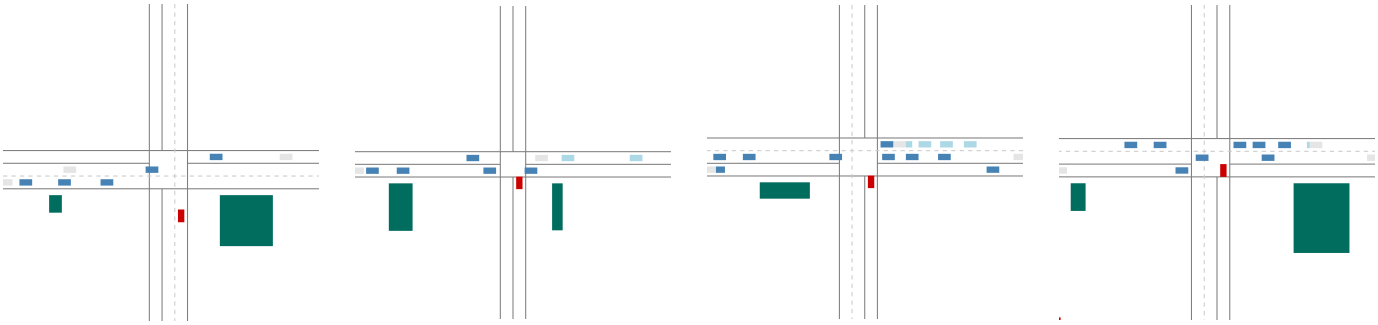

Figure 4: Examples of four different scenarios generated by our simulator for evaluating the proposed approach. For each intersection, number of lanes, location and size of obstacles are randomly generated. Red vehicle is the ego vehicle which should give way to the blue vehicles. Light blue cars are not visible for the ego vehicle due to sensor occlusion from obstacles (green areas) or from other vehicles. Gray rectangles show the phantom vehicles located at the maximum visible distance on each lane.

with random size and offset from intersection is generated in order to affect the sensor visibility. We assume maximum 70 meters visibility range and create the visibility polygon around vehicle position which is cut due to this obstacle (Figure 3). The position of stop line and also geometry of all intersection lanes are mapped to be used for situation representation as explained before. See [14] for some videos about the simulation environment and scenarios.

We also used our own abstract simulator for occluded intersections which is much faster and different intersections with multiple number of lanes and vehicles can be generated during training. Figure 4 depicts some examples of top view images from this simulator.

B. Impact of Deep-sets Architecture

Compared to the standard network architecture, the deep sets architecture has proven to be significantly more advantageous for our application. The deep sets based DQN has significantly smaller size and fewer parameters to learn comparing to the normal DQN and therefore it could be trained much faster. The highest reward of the standard architecture could be achieved with deep sets with almost half of the training time. At the same time, a policy could be learned through the deep sets architecture that is faster without taking a significantly higher risk (Figure 5). While with the standard architecture the training time was already longer even for five

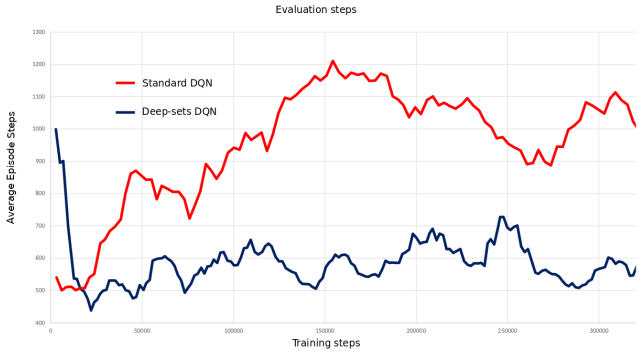

Figure 5: Comparing average evaluation steps for standard DQN and Deep-sets based DQN.

observed vehicles, a policy for up to eight considered vehicles was learned without any problems using deep sets. Due to the fact that deep sets structure does not assume fixed number of input vehicles, the learned policy could even perform reasonable when the number of vehicles at the intersection is much higher.

C. Return Distributions using IQN Approach

We utilized Implicit Quantile Networks (IQN) in order to learn return distributions in our intersection environment.

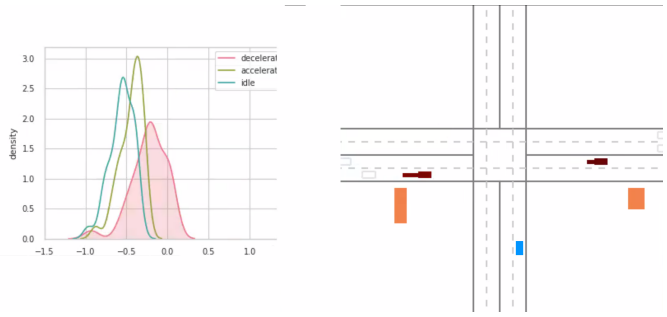

Figure 6: An example of learned distributions for the IQN agent (left image) at one state of the intersection (right image). Ego vehicle (blue) should decelerate in order to prevent a potential collision with the red car coming from the left side.

Figure 6 shows an example of learned distributions for each action at one specific situation. In this situation, the ego vehicle (blue) should decelerate since a fast vehicle from left side is entering the intersection and therefore the return for decelerate action is higher than other actions.

IV. Conclusion

In this paper we showed how deep-sets architecture can solve scalability issue and sensitivity to the input variations for the reinforcement learning agents used for automated driving at intersections. We proposed an architecture that does not depend on the number of vehicles and intersecting lanes at the intersection and can help to learn a policy which is more robust and less conservative but still safe. Using IQN approach in order to estimate return distributions instead of expected returns, we showed how it can help to learn risk aware policies which can prevent worst case outcomes using the learned distributions.

V. Acknowledgment

This research is accomplished within the project "UNICARagil" (FKZ 6EMO0287). We acknowledge the financial support for the project by the Federal Ministry of Education and Research of Germany (BMBF).

References

[1] David Isele, Reza Rahimi, Akansel Cosgun, Kaushik Subramanian, and Kikuo Fujimura. Navigating occluded intersections with autonomous vehicles using deep reinforcement learning. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2034–2039. IEEE, 2018.

[2] D. Isele, A. Nakhaei, and K. Fujimura. Safe reinforcement learning on autonomous vehicles. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–6, Oct 2018. doi: 10.1109/IROS.2018.8593420.

[3] Tommy Tram, Anton Jansson, Robin Grönberg, Mohammad Ali, and Jonas Sjöberg. Learning negotiating behavior between cars in intersections using deep q-learning. *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3169–3174, 2018.

[4] Maxime Bouton, Alireza Nakhaei, Kikuo Fujimura, and Mykel J Kochenderfer. Safe reinforcement learning with scene decomposition for navigating complex urban environments. In *2019 IEEE Intelligent Vehicles Symposium (IV)*, pages 1469–1476. IEEE, 2019.

[5] B. Mirchevska, C. Pek, M. Werling, M. Althoff, and J. Boedecker. High-level decision making for safe and reasonable autonomous lane changing using reinforcement learning. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 2156–2162, Nov 2018. doi: 10.1109/ITSC.2018.8569448.

[6] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *arXiv preprint arXiv:1610.03295*, 2016.

[7] Maria Huegle, Gabriel Kalweit, Branka Mirchevska, Moritz Werling, and Joschka Boedecker. Dynamic input for deep reinforcement learning in autonomous driving. *arXiv preprint arXiv:1907.10994*, 2019.

[8] Danial Kamran, Carlos Fernandez Lopez, Martin Lauer, and Christoph Stiller. Risk-aware high-level decisions for automated driving at occluded intersections with reinforcement learning. *arXiv preprint arXiv:2004.04450*, 2020.

[9] Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Russ R Salakhutdinov, and Alexander J Smola. Deep sets. In *Advances in neural information processing systems*, pages 3391–3401, 2017.

[10] Marc G Bellemare, Will Dabney, and Rémi Munos. A distributional perspective on reinforcement learning. *arXiv preprint arXiv:1707.06887*, 2017.

[11] Will Dabney, Mark Rowland, Marc G Bellemare, and Rémi Munos. Distributional reinforcement learning with quantile regression. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[12] Will Dabney, Georg Ostrovski, David Silver, and Rémi Munos. Implicit quantile networks for distributional reinforcement learning. *arXiv preprint arXiv:1806.06923*, 2018.

[13] Alexey Dosovitskiy, German Ros, Felipe Codevilla, Antonio Lopez, and Vladlen Koltun. CARLA: An open urban driving simulator. *Proceedings of the 1st Annual Conference on Robot Learning*, pages 1–16, 2017.

[14] Supplementary video file. https://www.dropbox.com/s/vnrjl0pro1uqw8w/rl_occlusion.avi?dl=0.