Minimizing Safety Interference for Safe and Comfortable Automated Driving with Distributional Reinforcement Learning

Danial Kamran[1], Tizian Engelgeh[2], Marvin Busch[2], Johannes Fischer[1] and Christoph Stiller[1]

Abstract— **Despite recent advances in reinforcement learning (RL), its application in safety critical domains like autonomous vehicles is still challenging. Although punishing RL agents for risky situations can help to learn safe policies, it may also lead to highly conservative behavior. In this paper, we propose a distributional RL framework in order to learn adaptive policies that can tune their level of conservativity at run-time based on the desired comfort and utility. Using a proactive safety verification approach, the proposed framework can guarantee that actions generated from RL are fail-safe according to the worst-case assumptions. Concurrently, the policy is encouraged to minimize safety interference and generate more comfortable behavior. We trained and evaluated the proposed approach and baseline policies using a high level simulator with a variety of randomized scenarios including several corner cases which rarely happen in reality but are very crucial. In light of our experiments, the behavior of policies learned using distributional RL can be adaptive at run-time and robust to the environment uncertainty. Quantitatively, the learned distributional RL agent drives in average 8 seconds faster than the normal DQN policy and requires 83% less safety interference compared to the rule-based policy with slightly increasing the average crossing time. We also study sensitivity of the learned policy in environments with higher perception noise and show that our algorithm learns policies that can still drive reliable when the perception noise is two times higher than the training configuration for automated merging and crossing at occluded intersections.**

I. INTRODUCTION

Reinforcement learning (RL) has gained more attention recently in order to solve complex decision making problems in robotics. Specifically for self-driving vehicles, long term optimal policies are learned using this approach for multiple scenarios such as lane changing or merging in highways [1]–[3] or yielding at unsignalized intersections [4]–[8]. Benefitting from the high representational power provided by neural networks to learn the future return of each action, learning-based policies can provide optimal behavior which is more generic and scalable compared to POMDP-based approaches like [9] and also more intelligent than rule-based approaches [10], [11].

Safety is one of the most important challenges for learning-based policies in critical applications like automated driving. Although the learned policies are prevented to

[1]Authors are with Institute of Measurement and Control Systems, Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany, {danial.kamran, johannes.fischer, stiller}@kit.edu
[2]Authors are students at Karlsruhe Institute of Technology (KIT), 76131 Karlsruhe, Germany, {tizian.engelgeh,marvin.busch}@student.kit.edu
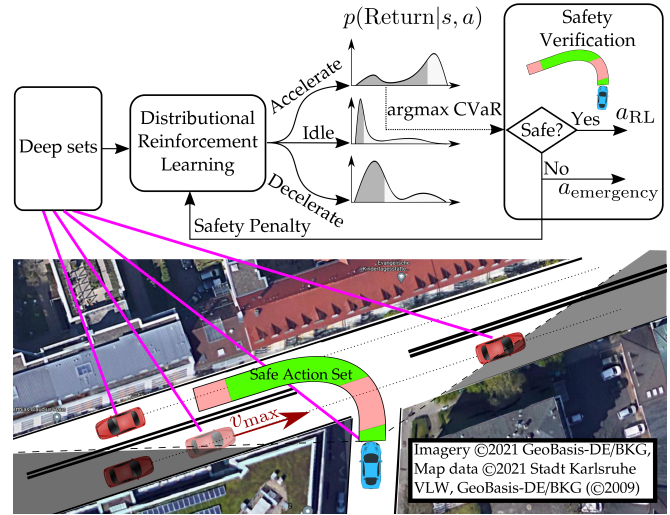
Fig. 1: Overview of the merging scenario and the proposed distributional reinforcement learning framework for automated navigation at similar occluded intersections (background image source: [12]).

generate risky behavior during training by receiving big punishments for having collisions [4], [5] or being in risky situations [6], [8], there is no guarantee that the learned policy is always safe after training. Another important challenge for the learning-based policies is to apply them in environments which are more challenging than the training environment due to higher sensor noise in perception or more severe sensor occlusions. Although one can try to learn a policy which reduces the amount of risk to be still safe in more challenging scenarios like [8], such a policy often is too conservative when the uncertainty is high. This dilemma between more conservative but slower policies and risky but faster policies is nicely elaborated by Isele et. al in [6] where higher timeout punishments as part of the reward function resulted in a faster but more risky policy. The origin of this problem lies in the difficulty of shaping the reward to guarantee having safe policies in gradient based learning approaches as discussed in [2].

Another approach to address safety for RL policies is to utilize a safety layer which filters out unsafe actions as proposed in [1], [13], [14]. This safety layer must be easily verifiable without any black boxes such as deep neural networks inside its architecture which are hard to verify [15]. In [1] authors used safety constraints in order to prevent

unsafe lane changes from a RL agent. As the fail-safe action, the automated vehicle will keep the lane whenever lane change actions are unsafe and in order to prevent unnecessary lane changes, the RL agent will receive punishments for every lane change decision. In [13], the authors showed that safety verification can also help the RL agent to converge faster in addition to guaranteeing safety. They studied the effect of applying the safety layer before or after the RL agent as *preemptive* and *Post-Posed Shielding* mechanisms and also specifying penalties for every safety intervention on the performance of the learned policy.

In the context of automated driving, uncertainties due to perception noise or ambiguous behavior of other partici-pants need to be considered inside the safety verification constraints. However, there is a trade-off between the level of safety that is guaranteed and efficiency [16]. Assuming that always the worst case perception noise exists and other drivers are all distracted will result in unnecessary safety interruptions which actually prevent the RL agent to utilize its learning skills. On the other hand, one can increase the capability of intervention (e.g. deceleration with -10 m/s^2) and only prevent unsafe actions just before a hazardous event is going to happen which will result in rare interventions but with uncomfortable feelings for the passengers.

Our contribution is to address safety, scalability and comfort as the main intertwined challenges for automated driving under uncertainty. We introduce safe distributional RL which considers maximum uncertainty and capability inside safety verification layer. During training the RL agent is punished for every safety intervention similar to [13], [14]. However, the main difference is it learns distributions instead of expected values for each action return which helps to provide risk-aware policies that are able to adapt their conservativity according to the existing uncertainty in the environment. This feature makes the learned policy generic and applicable for high uncertainty levels that can rarely happen in realistic applications, for example when the sensor range is severely occluded due to a parked vehicle or it has high uncertainty.

After explaining preliminaries of our work in section II, we describe the proposed worst-case based safety verification layer in section III and the safe distributional RL framework in section IV. Finally, efficiency of the proposed method is compared with a regular RL and a rule-based agent as baseline policies in section V.

II. PRELIMINARIES

A. Reinforcement Learning

Reinforcement learning problems are typically formulated as a Markov Decision Process (MDP) $(\mathcal{S}, \mathcal{A}, P, R, \gamma)$ [17]. In this framework, the environment is described by a state $s_t \in \mathcal{S}$ and the agent chooses an action $a_t \in \mathcal{A}$ at each discrete time step t. The action selection process is described by a policy π which is a mapping from states to actions (or to distributions over actions). The distribution of the successor state s_{t+1} is defined by the transition model $s_{t+1} \sim P(s_t, a_t)$ based on the current state and chosen action. In every step

the agent receives a reward $r_t = R(s_t, a_t)$ and its goal is to maximize the cumulative discounted reward (also called return) $\sum_{t=0}^{\infty} \gamma^t r_t$, where future rewards are discounted by the factor $\gamma \in [0, 1)$.

In many real-world applications the agent does not have full knowledge of the environment's state but instead receives only noisy observations of some state variables. Such situations can be described by a Partially Observable Markov Decision Process (POMDP) $(\mathcal{S}, \mathcal{A}, \mathcal{O}, P, \mathcal{Z}, R, \gamma)$, where the agent receives an observation $o_t \in \mathcal{O}$ at each time step. The observation depends on the current state and chosen action and is distributed according to the observation model $o_t \sim \mathcal{Z}(s_t, a_t)$. The agent's goal is still to maximize the return but with the additional challenge of an unknown environment state. Since each observations contains only limited information on the true environment state, policies for a POMDP therefore also depend on previous actions and observations.

A famous approach to find the optimal RL policy is Q-learning [18] which tries to maximize the expected future return defined as:

$$Q^\pi(s_t, a_t) = \mathbb{E}_{s_i \sim P}\left[R(s_t, a_t) + \sum_{k=1}^{\infty} \gamma^k R(s_{t+k}, \pi(s_{t+k}))\right],$$

by choosing action a_t in state s_t and following policy π for the next states [19]. Using the Bellman equation [20], the optimal value function can be represented as:

$$Q^*(s_t, a_t) = \mathbb{E}_{s_i \sim P}\left[R(s_t, a_t) + \gamma \max_{a'} Q^*(s_{t+1}, a'))\right].$$

In Deep Q networks (DQN) [21], deep neural networks are utilized to learn the Q values for each state and action over samples from a replay buffer. We use DQN as one of baselines in our experiments.

B. Distributional Reinforcement Learning

Distributional Reinforcement Learning [22] tries to learn return distribution $Z^\pi(s_t, a_t) \stackrel{D}{=} R(s_t, a_t) + \sum_{k=1}^{\infty} \gamma^k R(s_{t+k}, \pi(s_{t+k}))$ instead of its expected value. Note that here $\stackrel{D}{=}$ indicates equality in distribution. The value distribution can be computed using dynamic programming based on the *distributional Bellmann equation* [22] :

$$Z^\pi(s, a) \stackrel{D}{=} R(s, a) + \gamma Z^\pi(S', A'), \qquad (1)$$

where S' and A' are distributed according to $P(.|s, a)$ and $\pi(.|s')$, respectively. This equation is shown in [22] to be a contraction in the Wasserstein metric.

Similar to the regular RL, the *Distributional Bellman optimality equation* is applied:

$$\mathcal{T}Z(s, a) \stackrel{D}{=} R(s, a) + \gamma Z(S', \arg\max_{a' \in \mathcal{A}} \mathbb{E}Z(S', a')),$$

with S' distributed according to $P(\cdot|s, a)$. In order to learn the optimal return distribution, Bellemare et al. parameter-ized it as a categorical distributions over a fixed set of equidistant points [22]. Their approach, called C51, mini-mizes the Kullback-Leibler divergence to the distributional
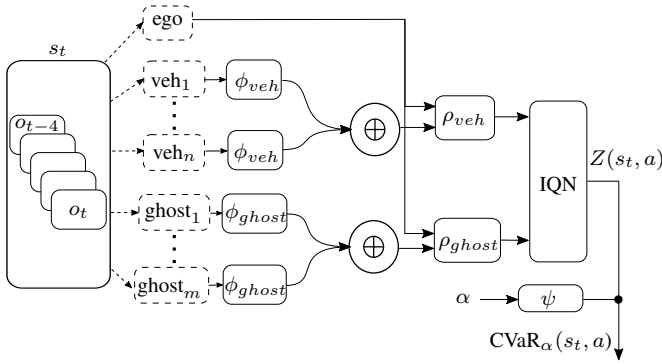
Fig. 2: Proposed Deep sets architecture for extracting permutation invariant and scalable features for the distributional RL network.

Bellman targets, which, however, was not a contraction in the Wasserstein metric. Later Dabney et al. proposed to learn return distributions through *Quantile Regression* (QR-DQN) on a fixed set of quantiles, minimizing the Wasserstein distance to the distributional Bellman targets [23]. In a newer approach, Dabney et al. introduced Implicit Quantile Networks (IQN) [24] to approximate the quantile function $F_Z^{-1}(\tau)$ for the random variable Z. Assuming $\tau \sim U([0,1])$, the return distribution can then be sampled from $F_Z^{-1}(\tau)$ as samples from implicitly defined return distribution. The main advantage of IQN is that any distortion risk measure $\beta : [0,1] \to [0,1]$ can be incorporated to compute distorted expectation of Z:

$$Q_\beta(s_t, a_t) = \mathbb{E}_{\tau \sim U([0,1])} \left[Z_{\beta(\tau)}(s_t, a_t) \right],$$

and the risk-sensitive greedy policy:

$$\pi_\beta(s_t) = \arg\max_{a \in \mathcal{A}} Q_\beta(s_t, a).$$

In this paper we utilize IQN implementation of distributional RL since it allows to explore risk-sensitive policies π_β during training which helps us to learn a family of risk-averse policies using only one neural network.

C. Observation and State Model

As proposed in [8], we can represent the whole situation at an occluded intersection using this matrix:

$$o_t = \begin{bmatrix} \overbrace{d_{\text{e,stl}}}^{ego} & \overbrace{d_1 \quad ... \quad d_n}^{vehicles} & \overbrace{d_{o_1} \quad ... \quad d_{o_m}}^{lanes} \\ v_{\text{e}} & v_1 \quad ... \quad v_n & v_{o_1} \quad ... \quad v_{o_m} \\ d_{\text{e,goal}} & d_{\text{e},1} \quad ... \quad d_{\text{e},n} & d_{\text{e},o_1} \quad ... \quad d_{\text{e},o_m} \end{bmatrix}^T \quad (2)$$

where v_{e} is the ego vehicle velocity and $d_{\text{e,stl}}$, $d_{\text{e,goal}}$ are its distance to the stop line and to the other side of the intersection. d_i, v_i are distance and velocity of every observable vehicle to their conflict zones and d_{o_i}, v_{o_i} are information for ghost vehicles indicating maximum visible distance and allowed velocity on every intersecting lane. Finally, $d_{\text{e},i}$ (d_{e,o_i}) represents the distance of the ego vehicle to every

(ghost) vehicle. We extend this representation for merging scenarios where we also observe the distance and velocity of the closest front vehicle on the ego vehicle lane.

Finally, we provide a k-Markov approximation [25] of the POMDP as input to the RL agent in order to enable Q-learning similar to [8]:

$$s_t = \begin{bmatrix} o_t & o_{t-1} & ... & o_{t-(k-1)} \end{bmatrix} \quad (3)$$

In this way the neural network and its training process become less complex compared to other implementations for POMDPs like [5] which utilize Long Short-Term Memory (LSTM) cells [26] to incorporate past information. In our experiments we supplied the last $k = 5$ observations to the network. Note that the direction of vehicles is specified by their velocity sign and their intention can be estimated from states history. Other traffic participants can also be added into this model like pedestrians or cyclist in order to provide more generic model. This representation is more efficient compared to the grid-based representations used in [4], [27] which require deep convolutional layers to extract useful features which are sensitive to the roads topology and irrelevant traffic participants.

D. Scalable Reinforcement Learning

The dimension of the state representation can exponentially grow for complex scenarios when the number of vehicles n or intersecting lanes m increase. Another challenge here is that permutation of input elements can change which may cause the network to have different reactions for the same scenario with different input permutations [28].

In order to address this problem, we use Deep sets [29] architectures that decouple the network size of machine learning algorithms from the number of input elements. Deep sets approach has already been applied to learn a lane change policy with DQN for highway scenarios in [28]. In this paper, we propose a Deep sets architecture for automated navigation at occluded merging and crossing scenarios (Figure 2). A representation is calculated for each type of the input element (real vehicle or ghost vehicles) with the ϕ_{real} and ϕ_{ghost} networks. After that, all features for each element type are combined with a permutation invariant operator, which we used sum of them. Finally, ρ_{real} and ρ_{ghost} networks extract fixed size features from the combination of each type of the input element with the ego vehicle state.

III. PROACTIVE SAFETY VERIFICATION

In order to verify safety, we propose a proactive safety verification approach which provides safe actions for every state $s \in \mathcal{S}$. We call it proactive while the worst-case scenario is always considered to prevent being in an unsafe state in the future.

A. Worst-Case Scenarios for Proactive Safety Verification

We define a set of worst-case scenarios $\mathcal{H}$ that can happen during automated driving at intersections:

1) On the intersecting lane l_i, an occluded vehicle is driving with velocity $v_{\max}$ at the closest occluded distance to the conflict zone.

2) On the intersecting lane l_i, an observed vehicle has an estimation error of $3\sigma_d$ and $3\sigma_v$ for its distance and velocity and accelerates with a_{max} to reach velocity of v_{max}.

3) On the ego lane l_{ego}, an observed vehicle in front of the ego vehicle has an estimation error of $3\sigma_d$ and $3\sigma_v$ for its distance and velocity and decelerates with a_{min} to reach zero velocity.

By assumptions in $\mathcal{H}$ we over-approximate the state of an occluded vehicle at intersecting lanes similar to [30] and for detected vehicles as well. In addition to [30], we over-approximate vehicles distance and velocity estimation error as $\mathcal{N}_a^t(\mu, \sigma_d^2)$ and $\mathcal{N}_a^t(\mu, \sigma_v^2)$ respectively which are denoted as truncated Gaussian distribution with support $[\mu - a\sigma, \mu + a\sigma]$. Therefore, the distance and velocity measurement errors are modeled with truncated Gaussian distributions $\mathcal{N}_3^t(d, \sigma_d^2)$ and $\mathcal{N}_3^t(v, \sigma_v^2)$, respectively, i.e. with a 3σ range around the true value.

B. Feasibility of Emergency Maneuvers for Safety Verification

For every intersecting lane l_i and also the ego vehicle lane l_{ego} inside the current state s, feasibility of executing one of the following two emergency maneuvers is evaluated to verify the state s as a proactive safe state (PSS) according to the worst-case assumptions $\mathcal{H}$:

- Emergency Stop Maneuver: Ego vehicle is able to stop at a distance d_{FS} bigger than SG_d before the conflict zone.
- Emergency Leave Maneuver: Ego vehicle is able to leave the intersection with a time headway t_{HW} bigger than SG_t.

Thus, PSS verification can be formulated as:

$$\text{PSS}(s) = \left(\bigwedge_{l_i \in s} d_{FS} > SG_d \right) \bigvee \left(\bigwedge_{l_i \in s} t_{HW} > SG_t \right), \quad (4)$$

with SG_d and SG_t the minimum safety gap required for stop and leave maneuvers. Note that here we validate safety according to all worst case assumptions in $\mathcal{H}$, i.e. if only one of them or all of them happens the situation should remain safe. We set $SG_d = SG_t = 0.5$ in our experiments which resulted in collision free maneuvers during training and evaluation of all policies.

C. Proactive Safety Verification of Actions in POMDP

We assume an agent selects action $a \in \mathcal{A}$ for an observation $o \in \mathcal{O}$ from our POMDP model. In order to verify safety of a, we need to make sure that it results in a proactive safe state even if the worst-case scenario happens. Therefore, we simulate the future state of the ego vehicle by executing a and the state of other vehicles with the worst-case assumptions in $\mathcal{H}$ until the next decision time. We call this simulated state as $\hat{s}_{\mathcal{H},o}^a$. If $\hat{s}_{\mathcal{H},o}^a$ is a PSS, then action $a \in \mathcal{A}$ is verified as a proactive safe action (PSA):

$$\text{PSA}(s, a) = \begin{cases} \text{True}, & \text{if } \text{PSS}(\hat{s}_{\mathcal{H},o}^a), \\ \text{False}, & \text{otherwise.} \end{cases} \quad (5)$$
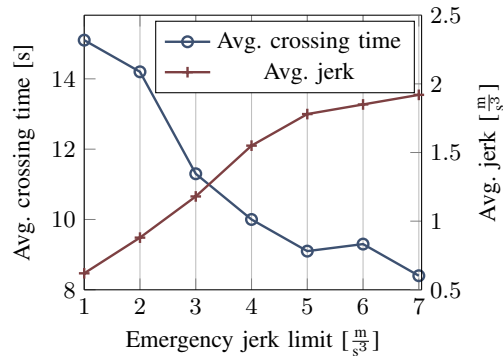

Fig. 3: Effect of increasing the jerk limit on the average crossing time and average jerk for the proactive safe policy.

The main advantage of the proposed worst-case proactive safety verification compared to reachability-based [31] and prediction-based approaches [6] for safety is that it only needs to predict the whole situation for one time step and only for the worst-case scenarios defined in $\mathcal{H}$ instead of all possible maneuvers for all participants in the whole future horizon. Since for each intersecting lane, the proposed safety verification approach only considers the closest vehicle to the conflict zone, it has computation complexity of $O(m + 1)$ where m is the number of intersecting lanes in the scenario. This complexity is much smaller than the safety verification strategy proposed in [6] which has complexity of $O(nT)$ where $n >> m$ is the number of agents and T is the time horizon considered for safety verification in that approach.

IV. LEARNING SAFE AND COMFORTABLE POLICIES FOR AUTOMATED DRIVING UNDER UNCERTAINTY

Similar to [2], we propose to apply safety as a hard constraint in the decision making problem trying to minimize other costs such as utility or comfort as soft constraints. For that, we can use the PSS concept in order to find all safe actions among the set of available actions in $\mathcal{A}$ as $\mathcal{A}_{\text{PSA}}$:

$$\mathcal{A}_{\text{PSA}}(s) = \{a \in \mathcal{A} | \text{PSA}(s, a) = \text{True}\}, \quad (6)$$

and search for the safe action with the lowest cost regarding comfort and utility. Here we consider actions as jerk commands applied to the automated vehicle similar to [32], [33] which help to provide comfortable maneuvers by moderating jerk. Therefore, we set possible discrete actions as $\mathcal{A} = \{-1.5\frac{m}{s^3}, 0.0\frac{m}{s^3}, 1.5\frac{m}{s^3}\}$.

A. Safe Rule-based Policy

We implemented a rule-based policy which selects the fastest safe action from $\mathcal{A}$ and in case of emergency selects an emergency action with the lowest jerk. The maximum emergency jerk limit can be increased which result in higher utility in cost of less comfortable drivings due to higher average jerk (Figure 3). On the other hand, when emergency jerk limit is reduced, the policy becomes slower, more comfortable and more conservative, since emergency maneuvers are limited.

B. Minimizing Safety Interference with Reinforcement Learning

RL, thanks to its ability to provide long-term optimal actions, can trade-off between utility and comfort based on the preferences defined in its reward function:

$$R(o,a) = \begin{cases} 1, & \text{if reach goal} \\ -\lambda, & \text{if } a \notin A_{\text{PSA}}(o) \\ 0, & \text{otherwise} \end{cases} \quad (7)$$

By such reward function, safety interventions are discouraged preventing situations that an uncomfortable emergency action with higher jerk instead of the unsafe RL action needs to be executed. The proposed framework has two main differences compared to other safe RL frameworks like [1], [13]. First, the emergency action is not necessarily a member of A_{RL}, therefore the training episode is terminated when a safety interference is required and the RL unsafe action will not be replaced in the replay buffer. Secondly, the agent receives a big punishment of $-\lambda$ whenever its action is not a member of A_{PSA} which is also not necessarily suggested in [13] and not used in [1] for safety interference.

One challenge here is choosing the safety interference punishment λ. Lazarus et. al. in [14] applied similar rewarding scheme for an autopilot system of the aircraft which punished the RL agent whenever the emergency controller had to be deployed. They showed choosing higher values for λ will encourage more conservative behavior, whereas faster policies with more interruptions can be expected for lower values of λ.

One may try to find the best value for λ, however, the learned policy generates proper behavior only for environments that have similar state transition probabilities to the training environment P. Moreover, due to the safety verification punishments which consider worst-case assumptions, RL either learns a super conservative policy or a fast policy with too many safety interventions. In our experiments, we learned a balanced policy by DQN with $\lambda = 1$ as a normal RL baseline.

C. Risk-aware Policies with Distributional Reinforcement Learning

The main problem with applying normal RL in environments under uncertainty is its risk-neutral characteristic that can not distinguish the amount of variance in actions' return and only considers their expected values. Therefore, one risky action with negative tail in its return but higher total expected value is always preferred to a safer (lower variance) action. For that, Tang et al. modeled the return for each action as a normal distribution and optimized an RL policy by learning the return distribution parameters (μ,σ) [3]. However, we believe that the return is a multimodal distribution and therefore we applied Implicit Quantile Networks (IQN) implementation [24] which approximates the quantile function that implicitly defines the return distribution. Moreover, utilizing the return distribution also allows to expand risk-neutral policies to risk-sensitive policies by applying distortion risk measures like the Conditional Value-at-Risk (CVaR) [34]:

$$\text{CVaR}_\alpha = \mathbb{E}\left[Z^\pi | Z^\pi \leq F_{Z^\pi}^{-1}(\alpha)\right], \quad (8)$$

where Z^π is the sum of discounted rewards under policy π and $F_{Z^\pi}^{-1}(\alpha)$ is its quantile function at $\alpha \in [0,1]$.

Therefore, we train an IQN agent with same reward function as DQN (Equation 7 with $\lambda = 1$) and tune the the risk-sensitivity of the policy using α at execution time without requiring to train multiple policies for different risk levels. In order to leverage the whole solution space over different risk-sensitive policies, α is uniformly sampled with $\alpha \sim U(0,1)$ at the beginning of each episode during training and applied to the IQN results (Figure 2).

V. RESULTS AND EVALUATIONS

A. Simulation and Policy Training

We train and evaluate different RL policies for automated merging and crossing at occluded intersections using our high level simulator which can simulate different randomized scenarios. Figure 5 shows an example scenario generated by our simulator. The location and size of obstacles (orange boxes) are generated randomly for each training episode causing some vehicles invisible for the ego vehicle. In order to generate realistic scenarios, every vehicle in the simulator (except the ego vehicle) has a random desired velocity selected by normal distribution with a randomly selected μ={6, 9, 12} and σ={2, 4, 6}. Each vehicle drives with the Intelligent Driver Model [35] controller with maximum acceleration $1\frac{m}{s^2}$, minimum deceleration $10\frac{m}{s^2}$ and comfortable deceleration $1.6\frac{m}{s^2}$ and keeps safety distance 2m and time headway 1.6s with front vehicles. However, the controller does not keep distance to the ego vehicle (when it drives into the intersection), thus a collision between a vehicle and the ego vehicle is possible.

At every step a new vehicle by probability of p_{new}={0.1, 0.4 or 0.7} is generated in the simulator. Vehicles are with probability p_{coop}={0.1, 0.4 or 0.7} cooperative and yield to the ego vehicle by setting their desired velocity to zero when the ego vehicle is close to the intersection which helps to simulate uncertainty about the intention of other drivers. Moreover, distance and velocity of vehicles are perturbed to simulate perception noise for the policy according to the truncated Gaussian distributions $\mathcal{N}_3^t(0, \sigma_d^2)$ and $\mathcal{N}_3^t(0, \sigma_v^2)$, respectively, where $\sigma_d = 1$m and $\sigma_v = 2\frac{m}{s}$ were fixed during training. σ_d and σ_v are linearly scaled according to the distance between the ego and other vehicle. For the ego vehicle, the RL policy generates a discrete jerk action from $\mathcal{A} = \{-1.5\frac{m}{s^3}, 0.0\frac{m}{s^3}, 1.5\frac{m}{s^3}\}$ every 0.3 second. RL policies are trained with more than 1000 intersection crossing and merging scenarios providing more than 3×10^5 training steps in total. Distributional RL policies have been trained with uniformly selected $\alpha \sim U([0,1])$ for CVaR calculation of the return distributions.
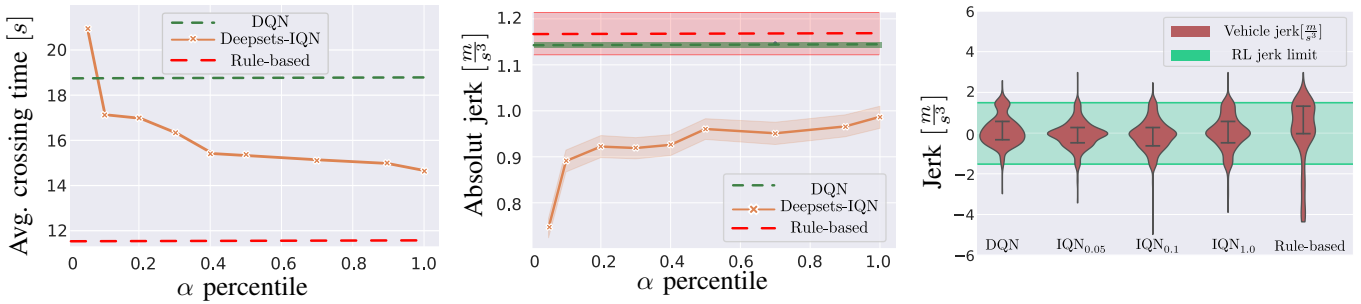
Fig. 4: Comparing performance of each policy on benchmark scenarios. **Left:** Average crossing time for each policy. IQN becomes faster and less conservative by higher α. Rule-based is the fastest and DQN is the slowest policy. Note that all policies are completely safe without any collision thanks to the safety layer. **Middle:** Average of vehicle absolute jerk and its confidence interval for each policy. By increasing α, IQN becomes less conservative (higher jerk). **Right:** Distribution of vehicle jerk while driving by each policy. Green interval shows RL jerk limits. Jerk values out of this interval are due to emergency interference and indicate uncomfortable driving.
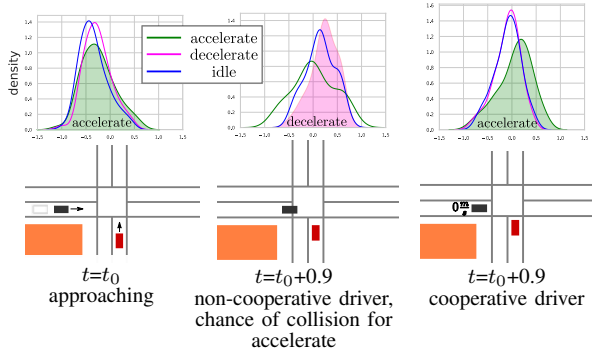

Fig. 5: **Left:** Learned return distributions for two scenarios with similar initial situation. Ego vehicle (red) follows the action with the highest CVaR that is shown with filled distribution. Gray ghost vehicle indicates maximum visible distance limited due to orange obstacles. **Middle:** Black vehicle is non-cooperative causing more negative return chance for the *accelerate* action. **Right:** Black vehicle is cooperative causing higher CVaR for *accelerate* action.

B. Evaluations

After training the RL agents, we evaluate their efficiency using 30 benchmark scenarios with random uncertainty configurations $\sigma_d \in \{0, 1m, 2m\}$, $\sigma_v = 2\sigma_d$ and p_{coop}={0.1, 0.4 or 0.7}. Note that, no collision with other vehicles happens during evaluations thanks to the safety verification layer. In case of an emergency situation, i.e. $a_{\text{RL}} \notin A_{\text{PSA}}$, an emergency maneuver from the safety layer with maximum emergency jerk limit of $5\frac{\text{m}}{\text{s}^3}$ is sent to the controller instead of the unsafe RL action. In order to compare the effect of interference applied for each policy, we define a metric for measuring the amount of applied emergency interference:

$$J_{\text{Interference}} = \frac{\sum a_{\text{emg}}^2}{N}, \tag{9}$$

where a_{emg} is the emergency jerk command applied to replace the unsafe action and N is the total number of evaluation episodes.

1) Drivers Intention Prediction: Since RL policies receive history of last 5 observations as their input state, they can predict the intention of other drivers and generate optimal actions based on that. Figure 5 shows examples of the learned return distributions by IQN agent for two similar scenarios with the only difference of having cooperative (yielding to the ego vehicle) and non-cooperative drivers. As it is visible, when the other vehicle is cooperative and reduces its velocity, the IQN policy generates higher return for $a = 1.5\frac{\text{m}}{\text{s}^3}$ action allowing the ego vehicle to enter into the intersection.

2) Comfort and Risk Sensitivity: Figure 4 compares the utility and conservativity of each RL policy compared with the safe rule-based policy. By increasing the α percentile, IQN policy becomes faster but less conservative in average. For lower α percentile, it has less absolute jerk (in the middle image) and lower jerks outside the RL jerk limit (in the right image) which indicates more comfortable maneuvers. DQN, shows the most conservative behavior with the highest crossing time. On the other hand, the rule-based policy has the highest amount of emergency interference as a risk neutral policy resulting in wider jerk distributions and showing uncomfortable maneuvers. Therefore, we can conclude that IQN can learn a *family* of adaptive policies which are not as highly conservative as DQN and also not as risk-neutral as the rule-based policy.

Same results can be concluded from Table I where we recorded the average crossing time and $J_{\text{Interference}}$ for all policies at different environment configurations. Here we only consider two IQN sub-policies, the ones with the fastest behavior and with the lowest interference cost (most comfortable). We evaluated the policy in 10 uniform samples of α and selected the best policy according to each metric. Note that finding the best α value could be done in a more automated approach but here we only wanted to compare the best behavior of the Deepset-IQN agent with other baseline agents and examine how much other metrics are sacrificed when the policy performs well in specific metric. In all three configurations considered in Table I, the rule-based policy

TABLE I: Evaluation of the IQN policies with best α for different metrics and their comparison with baselines for different environment configurations.

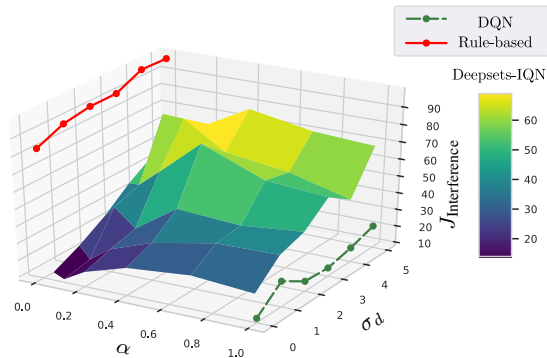| Environment Configuration		$\sigma_d = 0 \; p_{\text{coop}} = 0.3$		$\sigma_d = 2 \; p_{\text{coop}} = 0.3$		$\sigma_d = 2 \; p_{\text{coop}} = 0.7$	
Policy	Metric for optimal α	Time	$J_{\text{Interference}}$	Time	$J_{\text{Interference}}$	Time	$J_{\text{Interference}}$
Deepset-IQN	Speed	9.80	25.03	8.80	43.09	9.94	33.25
	Comfort	11.73	**10.00**	11.08	25.50	12.16	**18.83**
DQN	——	20.00	10.79	21.40	**14.17**	21.25	20.10
Rule-based	——	**7.69**	83.59	**7.75**	92.29	**7.74**	103.05

Fig. 6: Safety interference cost applied to the IQN policy for different environment noise levels and its comparison with baselines.

is the fastest one, however, it is also the worst policy in terms of comfort. The DQN, on the other hand, always has low interference cost, but drives more than two times slower than the others. The IQN trades of between average crossing time and interference cost by showing relatively fast behavior which requires 3.6 to 8 times lower interference than the rule-based policy depending on the environment configuration.

3) Evaluation under Higher Perception Uncertainties: In addition to the initial noise levels, we studied the sensitivity of each policy to higher amounts of noise in the environment. For that, we evaluated the RL and rule-based policies for 5 different noise levels $\sigma_d \in \{0, 1, 2, 3, 4, 5\}$ (in meters) and $\sigma_v = 2\sigma_d$ (in $\frac{\text{m}}{\text{s}}$). The results are depicted in Figure 6. As it is visible, the IQN policy has lower interference cost for lower α and also lower noise levels. For higher noise levels, it requires more interference which can be reduced by decreasing CVaR α. The rule-based policy always has high cost and DQN has low cost independent of the amount of noise existed in the environment. Here DQN seems to be a comfortable policy, however it is the worst policy in case of utility as discussed before.

VI. CONCLUSIONS AND FUTURE WORK

We proposed a proactive safety verification approach to validate the safety of actions with the goal of preventing unsafe situations in critical applications like automated driving. In order to minimize uncomfortable safety interference, we used reinforcement learning to learn policies that are punished for resulting in states where an emergency maneuver is necessary. We showed how IQN agent can mitigate the conservative behavior existing in DQN policies

by learning return distributions which provide policies that can be tuned adaptively after training in order to become more comfortable or less conservative. According to our experiments, the learned distributional policy requires less safety interference in noisy environments comparing to a rule-based safe policy even when the noise level is higher than the training configurations.

For future works, one can propose a better conservativity tuning approach in order to learn policies that automatically tune their risk sensitivity (α for the IQN policy) at runtime based on the observed uncertainty or the required risk sensitivity in the environment.

VII. ACKNOWLEDGMENT

REFERENCES

[1] 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)*, 2018, pp. 2156–2162.

[2] S. Shalev-Shwartz, S. Shammah, and A. Shashua, "Safe, multi-agent, reinforcement learning for autonomous driving," *arXiv preprint arXiv:1610.03295*, 2016.

[3] Y. C. Tang, J. Zhang, and R. Salakhutdinov, "Worst cases policy gradients," in *Conference on Robot Learning*, 2020, pp. 1078–1093.

[4] 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)*, 2018, pp. 2034–2039.

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

[6] 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)*, 2018, pp. 1–6.

[7] M. Bouton, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Safe reinforcement learning with scene decomposition for navigating complex urban environments," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, IEEE, 2019, pp. 1469–1476.

[8] D. Kamran, C. F. Lopez, M. Lauer, and C. Stiller, "Risk-aware high-level decisions for automated driving at occluded intersections with reinforcement learning," *arXiv preprint arXiv:2004.04450*, 2020.

[9] C. Hubmann, N. Quetschlich, J. Schulz, J. Bernhard, D. Althoff, and C. Stiller, "A pomdp maneuver planner for occlusions in urban scenarios," in *2019 IEEE Intelligent Vehicles Symposium (IV)*, 2019, pp. 2172–2179.

[10] R. van der horst and J. Hogema, "Time-to-collision and collision avoidance systems," Jan. 1994.

[11] J. Ziegler, P. Bender, M. Schreiber, H. Lategahn, T. Strauss, C. Stiller, *et al.*, "Making Bertha Drive—An Autonomous Journey on a Historic Route," *IEEE Intelligent Transportation Systems Magazine*, vol. 6, no. 2, pp. 8–20, 2014.

[12] *Google maps, by google.* https://goo.gl/maps/xbd5UvZpNACHfedF7.

[13] M. Alshiekh, R. Bloem, R. Ehlers, B. Könighofer, S. Niekum, and U. Topcu, "Safe reinforcement learning via shielding," *arXiv preprint arXiv:1708.08611*, 2017.

[14] C. Lazarus, J. G. Lopez, and M. J. Kochenderfer, "Runtime safety assurance using reinforcement learning," in *2020 AIAA/IEEE 39th Digital Avionics Systems Conference (DASC)*, IEEE, 2020, pp. 1–9.

[15] C. Liu, T. Arnon, C. Lazarus, C. Barrett, and M. J. Kochenderfer, "Algorithms for verifying deep neural networks," *arXiv preprint arXiv:1903.06758*, 2019.

[16] L. Sha *et al.*, "Using simplicity to control complexity," *IEEE Software*, vol. 18, no. 4, pp. 20–28, 2001.

[17] R. A. Howard, "Dynamic programming and markov processes.," 1960.

[18] C. J.C. H. Watkins and P. Dayan, "Q-learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.

[19] R. S. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. The MIT Press, 2018.

[20] R. Bellman, "Dynamic programming," *Science*, vol. 153, no. 3731, pp. 34–37, 1966.

[21] V. Mnih and D. Silver, "Playing Atari with Deep Reinforcement Learning," 2013. arXiv: 1312.5602.

[22] M. G. Bellemare, W. Dabney, and R. Munos, "A distributional perspective on reinforcement learning," in *International Conference on Machine Learning*, PMLR, 2017, pp. 449–458.

[23] W. Dabney, M. Rowland, M. Bellemare, and R. Munos, "Distributional reinforcement learning with quantile regression," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 32, 2018.

[24] W. Dabney, G. Ostrovski, D. Silver, and R. Munos, "Implicit quantile networks for distributional rein-forcement learning," in *International Conference on Machine Learning*, 2018, pp. 1096–1105.

[25] M. Bouton, K. Julian, A. Nakhaei, K. Fujimura, and M. J. Kochenderfer, "Utility decomposition with deep corrections for scalable planning under uncertainty," in *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems*, 2018, pp. 462–469.

[26] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, no. 8, pp. 1735–1780, 1997.

[27] D. Kamran, J. Zhu, and M. Lauer, "Learning path tracking for real car-like mobile robots from simulation," in *2019 European Conference on Mobile Robots (ECMR)*, IEEE, 2019, pp. 1–6.

[28] M. Huegle, G. Kalweit, B. Mirchevska, M. Werling, and J. Boedecker, "Dynamic input for deep reinforcement learning in autonomous driving," in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, IEEE, 2019, pp. 7566–7573.

[29] M. Zaheer, S. Kottur, S. Ravanbakhsh, B. Poczos, R. R. Salakhutdinov, and A. J. Smola, "Deep sets," in *Advances in neural information processing systems*, 2017, pp. 3391–3401.

[30] P. F. Orzechowski, A. Meyer, and M. Lauer, "Tackling occlusions & limited sensor range with set-based safety verification," in *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, IEEE, 2018, pp. 1729–1736.

[31] M. Althoff and J. M. Dolan, "Online verification of automated road vehicles using reachability analysis," *IEEE Transactions on Robotics*, vol. 30, no. 4, pp. 903–918, 2014.

[32] M. Werling, J. Ziegler, S. Kammel, and S. Thrun, "Optimal trajectory generation for dynamic street scenarios in a frenet frame," in *2010 IEEE International Conference on Robotics and Automation*, IEEE, 2010, pp. 987–993.

[33] J. Müller and M. Buchholz, "A risk and comfort optimizing motion planning scheme for merging scenarios," in *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, IEEE, 2019, pp. 3155–3161.

[34] R. T. Rockafellar, S. Uryasev, *et al.*, "Optimization of conditional value-at-risk," *Journal of risk*, vol. 2, pp. 21–42, 2000.

[35] M. Treiber, A. Hennecke, and D. Helbing, "Congested traffic states in empirical observations and micro-scopic simulations," *Physical review E*, vol. 62, no. 2, p. 1805, 2000.