Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnLine 55-15 (2022) 105–110

Automating Vehicles by Risk-Averse Preview-based Q-Learning Algorithm [⋆]

Majid Mazouchi [*] Subramanya Nageshrao [**]
Hamidreza Modares [*]

[*] Michigan State University, East Lansing, MI, 48863, USA, (e-mail: mazouchi@msu.edu; modaresh@msu.edu).
[**] Ford Research and Innovation Center, Ford Motor Company, Palo Alto, CA 94304, USA, (e-mail: snageshr@ford.com).

Abstract: A risk-averse preview-based Q-learning planner is presented for navigation of autonomous vehicles. To this end, the multi-lane road ahead of a vehicle is represented by a finite-state non-stationary Markov decision process (MDP). A sampling-based risk-averse preview-based Q-learning algorithm is finally developed that generates samples using the preview information and reward function to learn risk-averse optimal planning strategies without actual interaction with the environment. The risk factor is imposed on the objective function to avoid fluctuation of the Q values, which can jeopardize the vehicle's safety and/or performance. Theoretical results are provided to bound the number of samples required to guarantee ϵ-optimal planning with a high probability. Finally, to verify the efficiency of the presented algorithm, its implementation on highway driving of an autonomous vehicle in a varying traffic density is considered.

Keywords: Risk-averse, Q-learning, log-expected-exponential Bellman inequality.

1. INTRODUCTION

There are generally two opposite and yet intertwined requirements that an autonomous vehicle must satisfy: vehicle's performance and its safety. Vehicle's performance captures the cost incurred in achieving its goals (e.g., energy consumption). Vehicle's safety requirements impose constraints on its maneuvers to assure that its trajectories never enter the unsafe set while it traverses towards its goals. The conflict between safety and performance can arise as scenarios develop. This is because operating with maximum performance increases the vehicle's risk of getting into the unsafe set. The standard practice typically designs plans that optimize the expected value of vehicle's performance under uncertainties while ignoring its variance. However, while optimizing the expected value can lead to a good performance on average, some trajectories of the vehicle that are realized from some uncertainty realizations can significantly oscillate around the desired trajectory and thus increase the risk of getting into the unsafe set.

Since many sources of uncertainty in autonomous vehicles are aleatory (i.e., random), it is reasonable to design a planner to assure that the cost of its implementation is acceptable for all system trajectories realized from (uncertain) probability distributions of the random sources. Risk-averse strategies have been widely used for control of stochastic systems to avoid performance fluctuation and to reduce the risk of low probability but catastrophic events Hakobyan et al. (2019); Ramón Medina et al. (2012); Lin et al. (2018); Ratliff and Mazumdar (2020).

Reinforcement learning (RL) Sutton and Barto (2018), as the main tool for solving sequential decision-making problems under epistemic uncertainties, has been widely leveraged to learn an optimal control policy for uncertain systems. To account for both aleatory and epistemic uncertainties, risk-averse RL algorithms have been considered to solve risk-averse stochastic optimal control (RASOC) problems in Mihatsch and Neuneier (2002); Borkar (2002); Mazumdar et al. (2017), mainly for Markov Decision Processes (MDPs). Convex analytic approaches are developed in Haskell and Jain (2015, 2013) for risk-aware MDPs. In Shen et al. (2013), the authors present iterative risk measures which only depend on the current state, rather than on the whole history. In Mihatsch and Neuneier (2002); Borkar (2002), the authors investigate a risk-sensitive RL algorithm by applying utility functions to the temporal difference error. The most widely used RL algorithm for the RASOC problem is the policy gradient (PG) method Sutton and Barto (2018). PG has been utilized to learn the solution to VaR setting Geibel and Wysotzki (2005); Mazumdar et al. (2017), and entropic utility setting Dvijotham et al. (2014). However, PG methods generally result in non-convex optimization problems and are sensitive to fixed gradient step size, an inappropriate choice of which can easily make it divergent. Moreover, existing results on risk-averse RL require online interaction with the environment to learn about the value functions or their gradients. It is, however, highly desired for safety-critical systems to design risk-averse RL algorithms that can learn a policy using only available data collected from applying possibly several unknown control policies to the system.

[⋆] This work was supported by Ford Motor Company-Michigan State University Alliance.

In this paper, we design a risk-averse high-level planner for the navigation of autonomous vehicles between lanes around static and moving obstacles. A hierarchical control structure is leveraged: A risk-averse high-level planner chooses a sequence of waypoints and their corresponding high-level actions and a low-level controller drives the autonomous vehicle to assigned waypoints. From the high-level planner's perspective, the multi-lane road ahead of a vehicle is represented by a finite-state non-stationary MDP. A convex program-based preview-based Q-learning planner is then developed that turns the preview probabilistic reward information into a non-iterative risk-averse planning engine. The optimizing-based risk-averse Q-learning requires sampling state-action-reward tuples from the MDP. A great advantage of the presented approach is that these samples can be generated using the preview reward function without actual interaction with the environment. The developed scheme is evaluated for a case study of risk-averse autonomous vehicle navigation between lanes around obstacles.

Notation. Throughout this paper, we use $\mathbb{R}$, $\mathbb{R}_{\geq 0}$, $\mathbb{N}$ and to denote the sets of real numbers, non-negative real numbers, and non-negative integers, respectively. $\mathbb{R}^n$ and $\mathbb{R}^{n\times m}$ denote the n dimensional real vector space, and the $n \times m$ real matrix space, respectively. For the given variables x and y, $\langle x, y\rangle$ denotes the inner product of x and y. For a random variable X, $\mathbb{E}(X)$ denotes the expectation and $\operatorname{Var}(X) = \mathbb{E}\left(X^2\right) - \mathbb{E}(X)^2$ refers to the variance of X, respectively. exp(.) is an exponential operator such that $\exp(x) = e^x$. For convenience, $\mathbb{E}_\zeta$ refers to the expectation over the random variable ζ.

2. PROBLEM DESCRIPTION AND FORMULATION

To perform planning in highways, which are actually continuous-state domains, it has become common practice to use discrete planners by uniformly discretizing dimensions of the problem Mueller (2017). Grid-world Markov Decision Processes (MDPs) are the most common discrete models used for planning. However, planning over a long horizon by finding a policy for the entire domain simultaneously leads to an exponential growth in problem size as the number of states of the MDP increases. To sidestep this problem, the environment (i.e., the ground MDP) is decomposed into a series of grid-world MDPs with a finite number of cells in which the local reward and transition functions are evolved in discrete-time fashion. Therefore, at given predefined discrete-times $\tau_{env} > 0$, the autonomous vehicle (AV) senses only its neighborhood, i.e., a small part of the highway, and then updates the MDP grid-world (environment) through updating the local reward and transition functions of each cell. It is assumed that the changes on the behavior of other participants and the highway conditions and geometry are insignificant in a small-time horizon τ_{env}, and thus the MDP grid-world is assumed to be fixed and stationary for each τ_{env} period of time.

As scenarios develop, the local rewards corresponding to each cell in the environment will be evolved and updated, transitioning the vehicle into a new MDP. The reward function will be updated based on goal-reaching objective and safety objective. Depending on the risk level safety state of each of the $m \times n$ cells in the environment (i.e., the discretized finite portion of the highway around the ego-vehicle) is categorized into the following discrete states $\mathcal{Z} = \{un, sa, hr, lr, tg, cp\}$:

(1) Unsafe $\{un\}$– This kind of state represents the places in the environment that represents failure (i.e., they are occupied with any type of obstacles) or will eventually fail.
(2) Safe $\{sa\}$- This kind of state represents the places in the environment that are not currently failed and there are control actions to avoid them from getting into failure states.
(3) Highly Risky $\{hr\}$- This kind of state represents the places in the environment for which there is a significant chance that the safety mode cannot avoid crashing, i.e., all trajectories starting from these states will end in failure with significant probability.
(4) Low Risky $\{lr\}$- This kind of state represents the places in the environment for which safe mode can avoid crashing but some possible low-risk actions such as hard braking or sudden change of the direction are inevitable.
(5) Terminal Goal $\{tg\}$- This kind of state represents the terminal reaching state in the environment.
(6) Current Position $\{cp\}$- This kind of state represents the current position of the AV in the environment.

We now formally define and distinguish between the ground MDP $\mathcal{M}$ global MDP composed of a series of fixed local MDPs, i.e., $\mathcal{M} = \{\mu_1, \mu_2, ..., \mu_{N_T}\}$. The ground MDP captures the transition between local MDPs, which are then leveraged by the planner to perform re-planning. The safety state of MPD cells and consequently the reward function of the ground MDP changes from one MDP to another, but remains fixed for each local MDP from the planner's perspective. Therefore, the state of each local MDP is limited to the number of cells.

Definition 1. **(Environment ground MDP)** The finite-state MDP $\mathcal{M}$ for the high-level planner consists of a 5-tuple $\mathcal{M} : (\mathcal{S}_G, \mathcal{E}, \mathcal{A}_p, P_k, R_k)$, where:

- $\mathcal{S}_G$ is the global state space for the ground MDP with cardinality $|\mathcal{S}_G| = [6]^{m\times n}$, as each cell of each MDP can take 6 different safety states.
- $\mathcal{E} \equiv \{1, 2, \ldots, N_T\}$ is the set of transitory epochs in which local MDPs change, with $N_T \leq +\infty$.
- A set of global actions $a_p \in \mathcal{A}_p$ according to a single or sequence of the planner function ${P_k}^{a_p}(S, S') := P_k\left(S' \mid S, a_p\right)$ or a change in behavior of other traffic participants that changes the MPD.
- Transition probabilities ${P_k}^{a_p}(S, S') := P_k\left(S' \mid S, a_p\right)$ which denotes the probability of a transition to the next global state S' when taking action a_p at transitory epoch k in the global state S.
- The probabilistic reward function ${R_k}^{a_p}(S, S') := R_k(S' \mid S, a_p)$ which denotes the probability of changing the MDP cells reward values associated to the transition from the current global state $S \in \mathcal{S}_G$ to the next global state $S' \in \mathcal{S}_G$ when a global action a_p is taken.

Based on Definition 1, between each consecutive transition epochs, i.e., τ_{env}, the environment of AV (i.e., $\mathcal{M}$) can be seen as a stationary and fixed grid-world MDP. We now define more formally the environment state space for the k-th local MDP, μ_i over a time horizon $(k+1)\tau_{env} - k\tau_{env}$ from the high-level planner's perspective.

Definition 2. **(k-th local MDP)** The finite-state local MDP μ_k consists of a 5-tuple $\mu_k : (S_k, \mathcal{A}, \bar{P}^a_k, \bar{R}_k, \gamma)$, where:

- Finite state space S_k with cardinality $|S_k| = m \times n$ where S_k consists of $m \times n$ cell states $s \in \{cell^j_i : i = 1, ..., m, j = 1, ..., n\}$.
- A finite set of available high-level actions $a \in \mathcal{A}$.
- Taking action $a \in \mathcal{A}$ and transitioning from the current state $s \in S_k$ and the next state $s' \in S_k$ incurs the reward received value based on the stationary reward distribution $\bar{R}_k(s' \mid s, a)$.
- The local transition probabilities $\bar{P}^a_k(s, s') := \bar{P}_k(s' \mid s, a)$ which denotes the probability of a transition to state s' when taking action $a := \pi_k(s)$ in the state s according to a policy π_k .
- $\gamma \in (0, 1]$ is the discount factor for future rewards (a reward received after $l \in \mathbb{N}$ steps is weighted by γ^l);

It is assumed here for simplicity that the transition probabilities $\bar{P}_k(s' \mid s, \pi_k(s))$ of the local MDP are known beforehand and not changing over time, i.e., $\bar{P}_k(s' \mid s, \pi_k(s)) = \bar{P}(s' \mid s, \pi_k(s))$ for $k = 1, ..., N_T$. This is a reasonable assumption since if the higher-level planner commands to move from one waypoint to another using its high-level actions, the low-level controller aims to do so. The probability of transitions can be learned from the success rate of the low-level controller in performing its high-level actions. Therefore, as MDPs change in the ground MDP, only their reward functions are assumed to change. Moreover, it is also assumed here that there exist a risk-assessment (RAU) and feasibility check unit (FCU) modules which, respectively, provides the preview reward function for the current environment and continuously monitor the feasibility of the generated desired reference trajectory for the next $n\tau_{env}$ seconds of the travel (as discussed in the extended online version of this paper Mazouchi et al. (2021)).

Let $g^k_a(s)$ be the prediction of the expected distribution reward received when taking action a at state s and transition epoch k as $g^k_a(s) = \mathbb{E}_{s' \sim \bar{P}(\cdot|s,a)}\left[\bar{R}_k(s, a, s')\right]$ which is provided at each transition epoch by the RAU module. The planner will learn a policy using stationary and fixed grid-world MDPs which are called imaginary MDP from now on. This name comes from the fact that MDPs of the ground MDP are constructed based on preview information which in turn provided imagine reward functions, which can be different from the actual MDP. Now, we define the k-th imaginary MDP as $\mu^I_k : (\mathcal{S}_k, \mathcal{A}, \bar{P}^a, g^k_a(s), \gamma)$ which is used by the high-level planner. To account for the difference between the actual MDP and the imagined one, a feasibility unit is introduced to check for infeasible plans due to this difference and mitigate possible risks by re-planning.

The objective of the high-level risk-averse planner is to learn the function $pl(\mathcal{S}_k) : \mathcal{S}_G \to \mathcal{A}^N$ by optimizing a risk-averse objective function with the effective horizon of N for the imaginary MDP μ^I_k. That is, the planner learns a sequence of high-level actions $a_1, ..., a_N$ for the local MDP μ^I_k, providing the planing policy $\{\Pi_k : k \in \mathcal{E}\}$ where $\Pi_k = \{a_l = \pi^k(s_l) : l = 0, ..., N\}$.

3. A SAMPLING-BASED CONVEX PROGRAM FOR RISK-AVERSE PREVIEW-BASED LEARNING

Consider the imaginary MDP μ^I_k defined based on Definition 2. The process begins at imaginary time $l = 0$. A policy $\pi_k : S_k \to \mathcal{A}$ for the high-level planner is a mapping from MDP states S_k to high-level actions $\mathcal{A}$. Given the imaginary MDP μ^I_k, the risk-averse planner's goal is to learn a sequence of high-level actions to reach a cell state with discrete states type terminal goal, i.e., tg, state from an initial state while optimizing a risk-informed cost function.

Given the finite set of cell states $s \in S_k$ and the set of actions $\Pi_k = \{a_l = \pi^k(s_l) : l = 0, ..., n\}$ which are generated by the policy π_k, and the reward function $g^k_a(s)$, all defined in the imaginary MDP μ^I_k, an infinite-horizon discounted cost is first defined in the form of

$$\mathcal{L}^{\pi_k}(s_0) = \sum_{l=0}^{\infty} \gamma^l g_{\pi^k}(s_l), \tag{1}$$

where $g_{\pi^k}(s)$ is used as shorthand for $g_{\pi^k(s)}(s)$. Note that even though the cost is defined over an infinite horizon for simplicity of the analysis and design, the effective horizon of the planning, after which the reward can be ignored, is finite due to the discount factor and depends on the value of the discount factor. Moreover, safety and performance specifications are encoded in the reward function $g_{\pi^k}(s)$.

Since the probability transitions $\bar{P}(s' \mid s, \pi_k(s))$ and the reward function $g_{\pi^k}(s)$ provided by the RAU are probabilistic, optimizing the expected value of the cost function (1) is not good enough since it can lead to performance fluctuation and even unsafe behaviors. To account for the risk or variance of the performance, which provides high-confidence safety assurance, the entropic based risk-averse cost function J is used as

$$J_{\pi^k}(s) = \frac{1}{\alpha}\log\left(\mathbb{E}\Big[\exp\left(\alpha\mathcal{L}^{\pi^k}\right) \mid s_0 = s\Big]\right), \tag{2}$$

where $\alpha \in \mathbb{R}_{>0}$ is the risk-averse factor, and the subscript l is dropped for notational simplicity. One has,

$$J_{\pi^k}(s) \approx \mathbb{E}\left(\mathcal{L}^{\pi^k}(s)\right) + \frac{\alpha}{2}\mathrm{Var}\left(\mathcal{L}^{\pi^k}(s)\right). \tag{3}$$

Lemma 3. (Bäuerle and Jaśkiewicz (2018)) The risk-averse cost function (2) is continuous, non-decreasing, and convex.

Problem 1. (Entropic risk-averse optimal planning problem (ERAOP)) Let the reward function (1) be previewed by the RAU module based on preview information for an imaginary MDP μ^I_k. Design a preview-based optimal plan $\pi^*_k : S_k \to \mathcal{A}$ that minimizes the risk-averse cost function (2) for the resulting MPD μ^I_k. That is, find π^*_k that solves

$$J_{\pi^*_k}(s_0) = \inf_{\pi_k \in \mathcal{U}} J_{\pi_k}(s_0) = J^*_{\pi_k}(s_0) \tag{4}$$

using only preview information, where $\mathcal{U}$ is the set of feasible plan policies that assure boundness of the entropic cost.

For a given policy π_k and for the cost (2), define the value function as

$$V^{\pi_k}(s) := J_{\pi_k}(s) \tag{5}$$

To compute the value of each state-action pair (s, a), the Q-function associated with the action $a = \pi_k(s)$ is defined as

$$Q^{\pi_k}(s,a) := g_a^k(s) + \tfrac{1}{\alpha}\log(\mathbb{E}_{s'_a}[\exp(\alpha\gamma \sum_{s'_a\in S_k} \bar{P}^a_{s,s'}V^{\pi_k}(s'_a))]) \tag{6}$$

where $\bar{P}^a_{s,s'} = \bar{P}(s' \mid s,a)$ and $V^{\pi_k}(s) = Q^{\pi_k}(s,a)$. Defining the optimal Q-function $Q^*(s,a) := Q^{\pi_k{}^*}(s,a)$ gives the optimal Bellman equation

$$\begin{aligned}&Q^*(s,a) = g_{\pi_k}(s) + \inf_{a'\in\mathcal{A}}\Big\{\frac{1}{\alpha}\times\\ &\log\Big(\mathbb{E}_{s'_a}\Big[\exp\Big(\alpha\gamma\sum_{s'_a\in S_k}\bar{P}^a_{s,s'}V^{*\pi_k}(s'_a)\Big)\Big]\Big)\Big\}\end{aligned} \tag{7}$$

where a' denotes the next action under the optimal policy π_k^*. The optimal control policy is then given by

$$\pi_k^*(s) = \arg\min_{a\in\mathcal{A}} Q^*(s,a) \tag{8}$$

Using the optimal Q risk Bellman operator $\mathbb{T}$ as

$$\begin{aligned}&\mathbb{T}Q^*(s,a) := g_{\pi_k}(s) + \inf_{a'\in\mathcal{A}}\Big\{\frac{1}{\alpha}\times\\ &\log\Big(\mathbb{E}_{s'_a}\Big[\exp\Big(\alpha\gamma\sum_{s'_{\pi_k(s)}\in S_k}\bar{P}^{\pi_k(s)}_{s,s'}Q^*(s'_{\pi_k(s)},\pi_k(s'_{\pi_k(s)}))\Big)\Big]\Big)\Big\}\end{aligned} \tag{9}$$

the optimal Bellman equation becomes

$$Q^*(s,a) = \mathbb{T}Q^*(s,a) \tag{10}$$

The infimum operator makes the optimal Bellman operator hard to solve. Therefore, instead of directly solving it, the following easier-to-solve Bellman operator is introduced.

$$\begin{aligned}&\widehat{\mathbb{T}}Q^{\pi_k}(s,a) := g_a^k(s) + \frac{1}{\alpha}\times\\ &\log\Big(\mathbb{E}_{s'_a}\Big[\exp\Big(\alpha\gamma\sum_{s'_{\pi_k(s)}\in S_k}\bar{P}^{\pi_k(s)}_{s,s'}Q^{\pi_k}\big(s'_{\pi_k(s)},u(s'_{u(s)})\big)\Big)\Big)\Big]\end{aligned} \tag{11}$$

To learn a risk-averse planning solution for an environment that evolves as a series of MDPs, collecting data for every new MDP is required, which is time-consuming and not practical since a plan must be made before the decision horizon is finished. This is especially true for autonomous driving for which scenarios keep changing, which in turn change the MDP model.

It is shown in the following propositions that this operator is monotone and contractive.

Definition 4. Meyer (2000) Let $\mathcal{C}(S_k)$ be a complete space. An operator $\mathbb{T} : \mathcal{C}(S_k) \to \mathcal{C}(S_k)$ is monotone if

$$\langle \mathbb{T}v_1 - \mathbb{T}v_2, v_1 - v_2\rangle \geq 0, \quad \forall v_1, v_2 \in \mathcal{C}(S_k) \tag{12}$$

Proposition 5. The Bellman operator $\widehat{\mathbb{T}}$ in (11) is monotone.

Proof. See the extended online version Mazouchi et al. (2021, Proposition 1) for the proof. □

We now show that the Bellman operator with a reward function which is not necessarily bounded by a constant is a contraction mapping with respective to a weighted norm defined below.

Definition 6. Bertsekas (2018) Consider a weight function $w : S_k \to \mathbb{R}$ where S_k is a finite-dimensional vector space. Let $w(s) > 0, \forall s \in S_k$ and define the weighted sup-norm as

$$\mathcal{J}_w = \sup_{s\in S_k}\frac{|\mathcal{J}(s)|}{w(s)} \tag{13}$$

Assumption 1. There exists a constant $\rho \geq 0$ and a non-decreasing function $w : \mathbb{R} \to [1, \infty)$ such that

$$g_{\pi_k}(s) \leq \rho w(s), \quad \forall s \in S_k \tag{14}$$

where $w(s)$ satisfies

$$\sup_{a\in\mathcal{A}}\mathbb{E}_{s'_a} w(s'_a) \leq \Upsilon w(s), \forall s \in S_k \tag{15}$$

for some constant $\Upsilon \in \left(0, \frac{1}{\beta}\right)$.

Theorem 7. Under Assumption 1, the Bellman operator in (11) is a contraction mapping operator with respect to the weighted norm w. That is,

$$\left\|\widehat{\mathbb{T}}Q^{\pi_k^2}(s,a) - \widehat{\mathbb{T}}Q^{\pi_k^1}(s,a)\right\|_w \leq \Upsilon\gamma\left\|Q^{\pi_k^2}(s,a) - Q^{\pi_k^1}(s,a)\right\|_w \tag{16}$$

$\forall (s,a) \in S_k \times \mathcal{A}$ where $\Upsilon\gamma \leq 1$.

Proof. See the extended online version Mazouchi et al. (2021, Theorem 1) for the proof. □

Proposition 8. Let Q_0 be an arbitrary Q-function. Then, the following property hold.

$$Q^{\pi_k}(s,a) = \lim_{l\to\infty}\widehat{\mathbb{T}}^l Q_k(s,a) \tag{17}$$

when the actions are generated by a fixed policy π_k, i.e., $a = \pi_k(s)$.

Proof. The proof is omitted due to the limited space. □

Proposition 9. A Q-function that satisfies the Bellman inequality $Q \leq \widehat{\mathbb{T}}Q$ provides a lower bound to Q^{π_k}.

Proof. Considering $Q \leq \widehat{\mathbb{T}}Q$ and applying the operator $\widehat{\mathbb{T}}$ on its both sides repeatedly, and using the results of Proposition 8 yields

$$Q_k \leq \widehat{\mathbb{T}}Q_k \leq \widehat{\mathbb{T}}^2 Q_k \ldots . \leq \widehat{\mathbb{T}}^\infty Q_k = \lim_{l\to\infty}\widehat{\mathbb{T}}^l Q_k = Q^{\pi_k} \tag{18}$$

This completes the proof. □

Problem 2. (preview-based convex program)

$$\begin{aligned}&\max \quad c(s,a)Q(s,a)\\ &Q(s,a) \leq g_{\pi_k}(s) + \frac{1}{\alpha}\times\\ &\log\Big(\mathbb{E}_{s'_a}\Big[\exp\Big(\alpha\gamma\sum_{s'_{\pi_k(s)}\in S_k}\bar{P}^{\pi_k(s)}_{s,s'}Q\big(s'_{\pi_k(s)},\pi_k(s'_{\pi_k(s)})\big)\Big)\Big]\Big)\\ &\forall (s,a,a') \in S_k \times \mathcal{A} \times \mathcal{A}\end{aligned} \tag{19}$$

Corollary 10. The solution to the optimization problem (19) coincides with the optimal value function Q^{*k} in (7).

Proof. See the extended online version Mazouchi et al. (2021, Corollary 1) for the proof. □

After finding the approximate optimal function Q^{*k} from solving Problem 2, finding the greedy approximate optimal control policy by solving

$$\begin{aligned}&\pi_k^*(s) = \arg\min_{a\in\mathcal{A}}\Big\{g_{\pi_k}(s) + \frac{1}{\alpha}\times\\ &\log\Big(\mathbb{E}_{s'_a}\Big[\exp\Big(\alpha\gamma\sum_{s'_{\pi_k(s)}\in S_k}\bar{P}^{\pi_k(s)}_{s,s'}Q^*\big(s'_{\pi_k(s)},\pi_k(s'_{\pi_k(s)})\big)\Big)\Big]\Big)\Big\}\end{aligned} \tag{20}$$

Note that optimization is formed using only preview information and solved for the MDP ahead before the scenario actually becomes apparent. Algorithm 1 provides the details of the risk-averse preview-based Q-learning algorithm.

Algorithm 1 Convex program preview-based risk-averse Q-learning algorithm.

Result: The optimal risk-averse state-action values Q^k.
Data collection: Select different control policies $\{\pi_k^1, \pi_k^2, \ldots, \pi_k^\tau\}$. For each control policy j, $j = 1, ..., \tau$, generate state-action tuples $\mathcal{D}_j^k = \left\{\left(s^i, \pi_k^j\left(s^i\right), s^{i'}, \pi_k^j\left(s^{i'}\right)\right)\right\}_{i=1}^{N_k}$ randomly where $s^{i'}$ is the next state base on known $\bar{P}_{s^i, s^{i'}}^{\pi_k(s^i)}$ and previewed reward function. Now, create a database $\mathcal{D}^k = \{\cup \mathcal{D}_j^k\}$, $j = 1, ..., \tau$.
Function evaluation: Solve

$$\begin{aligned}
&\max \quad c(s,a)Q(s,a)\\
&Q(s^i,\pi_k(s^i)) \le g_{\pi_k}(s^i) + \frac{1}{\alpha}\log\Big(\frac{1}{\iota}\sum_{\kappa=1}^{\iota}\Big[\exp\Big(\alpha\gamma\times\\
&\sum_{s'_{\pi_k(s^i)}\in S_k} \bar{P}_{s^i,s^{i'}}^{\pi_k(s^i)} Q(s^{i'}_{\pi_k(s^i)}, \pi_k(s^{i'}_{\pi_k(s^i)}))\Big)\Big]\Big)\\
&\forall (s^i, \pi_k(s^i), \pi_k(s^{i'})) \in \mathcal{D}^k
\end{aligned} \tag{21}$$

Theorem 11. Assume that there exists a feasible plan for the imaginary MDP μ_k^I. Let the violation and confidence levels be chosen as ϵ and $\bar{\beta}$, respectively. Let the cardinality of $\mathcal{D}^k$ be $N_{\mathcal{D}^k}$ and the number of state-actions of the Q-function be N_Q. Let the samples be drawn independently and identically distributed according to a probability measure. Then, if $N_{\mathcal{D}^k} \ge \frac{1}{\epsilon}(N_Q + \log(\frac{1}{\bar{\beta}}))$ with probability at least $\bar{\beta}$, the solution of Algorithm 1 is a feasible solution and optimal solution for a subset of the state-action set $(S_k, \mathcal{A})$ which is of at least measure $(1-\epsilon)$.

Proof. Problem 2 is a convex optimization problem with finite number of state-action pairs. This is a special case of the convex optimization problem with infinite number of realizations considered in Campi et al. (2009). Inspired by Campi et al. (2009) and using scenario-based optimization, scenarios (states and actions here) can be sampled and the same proof as in Campi et al. (2009) can be leveraged to obtain the sample bound $N_{\mathcal{D}^k} \ge \frac{1}{\epsilon}(N_Q + \log(\frac{1}{\bar{\beta}}))$. □

4. SIMULATION

As a case study to show the validity of the proposed risk-averse planner scheme, a scenario of highway driving of an autonomous vehicle in a varying traffic density is considered here. The model dynamics of ego-vehicle is chosen as a single-track bicycle model vehicle given by $\dot{Z} = AZ + B\delta$, with $Z^T = \left[Y, \Psi, \alpha_T, \dot{\Psi}\right]^T$,

$$A = \begin{bmatrix} 0 & v_T & v_T & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -\dfrac{2K}{m_T v_T} & -\dfrac{m_T v_{T,0} + \frac{-0.2692K}{v_T}}{m_T v_T} \\ 0 & 0 & \dfrac{0.2692K}{I_T} & -\dfrac{a^2K + b^2K}{I_T v_T} \end{bmatrix}, B = \begin{bmatrix} 0 \\ 0 \\ \dfrac{K}{m_T v_T} \\ \dfrac{1.6154K}{I_T} \end{bmatrix} \tag{22}$$

where $Y(t)$ and $\delta(t)$ denote the vertical position and steering angle of the controlled vehicle along its lane, respectively. Moreover, Ψ and $\dot{\Psi}$ are the yaw and yaw rate, respectively, α_T is the vehicle sideslip angle, $\dot{X} = v_T + X(0)$ and $\dot{v}_T = 0$, where X and v_T are the longitudinal position of the controlled vehicle and vehicle velocity, respectively. Moreover, $m_T = 1300.0$kg, $I_T = 10^4$kg $\cdot$ m^2, $K = 91090$, $\alpha_{T,0} = -0.2rad$, $\mu = 0.8$, $\dot{\Psi}_0 = 0.7rad/s$, and $v_0 = 16.75m/s$, denote the total mass, moment of inertia, tires stiffness, initial vehicle sideslip angle, friction coefficient, initial yaw rate, and initial velocity, respectively. The ego-vehicle state vector is $S_a = [X, Y, \Psi, \alpha, \dot{\Psi}]^T \in \mathbb{R}^5$, its observable state vector is $S_o = \{X, Y\} \in \mathbb{R}^2$, and its control input is δ steering angle.

For the low-level controller, linear quadratic regulator (LQR) state feedback controller is utilized. To this end, given a planned trajectory $\mathcal{T}_p(t)$ based on the planned waypoints $(X_p, Y_p) \in \mathcal{T}_p(t)$ provided by high-level risk-averse Q-learning planner, the main objective of the low-level LQR state feedback controller is to follow $\mathcal{T}_p(t)$ and stabilize the ego-vehicle yaw, yaw rate, and side slip angle states, i.e., $s_v = [\Psi, \alpha, \dot{\Psi}]^T$. That is, $e(t) := ([Y - Y_p, \Psi, \alpha, \dot{\Psi}]^T)^2 \to 0$. To this aim, the LQR performance index is chosen as $\int_0^\infty \left(e^T\mathbf{Q}e + \delta^T\mathbf{R}\delta\right)\mathrm{d}t$ where $\mathbf{Q} = diag(3,1,1,1)$ and $\mathbf{R} = 1$. Note that $X - X_p$ is removed from $e(t)$ since $v_T = 0$ and the low-level controller control the lateral dynamics of the ego-vehicle.

The ego-vehicle and the vehicles 2, 3, and 4 speeds are 16.7 m/s, $2.6m/s$, $-2.6m/s$, and $-2.6m/s$ respectively. In performing closed-loop experiments, the centers of gravity of ego vehicle and other vehicles are initialized at $(x(0), y(0)) = (0,1)m$, $(x_1(0), y_1(0)) = (28, -4)m$, $(x_2(0), y_2(0)) = (56, 0)m$, $(x_3(0), y_3(0)) = (164.5, 6)m$, and $(x_4(0), y_4(0)) = (157.5, -6)m$.

The width and length of the highway are $22m$ and $203m$, respectively. We assume that the preview information is just available for the next eight seconds of travel. The environment is modeled as a 2-dimensional 11×39 grid with a total of 429 cells with width of $2m$ where each square forward represents the next 0.2 second of the travel (i.e., length of $3.5m$ since the ego-vehicle traveling with constant velocity of $16.75m/s$) and cells can have different characteristics. The dimensions of the ego-vehicle are considered as $2m \times 3.5m$, and the dimensions of the other vehicles are chosen as $4m \times 7m$. Each of these cells in the environment can be in one of the five discrete states $\mathcal{Z} = \{un, sa, hr, lr, tg, cp\}$. The edges of the highway occupies one square an each of three lanes occupies three squares. The finite set of high-level actions is $\mathcal{A} = \{1, 2, 3\}$ where the high-level actions are defined as: $\{1\}$: Go forward to the next state based on $\mathbf{Sec}_1$; $\{2\}$: Go forward to the next state based on $\mathbf{Sec}_2$; $\{3\}$: Go forward to the next state based on $\mathbf{Sec}_3$. The simulation time is 12 seconds, and the following values are used: $\gamma = 0.3$, $\tau_{env} \simeq 0.2sec$, $n = 39$, and $N_k = 1000$, $\tau = 10$, $M = -10000$, $\tau_l = 10000$, $\tau h = 0$, $\sigma = 1$, $\bar{\tau}_l = 10$, $\bar{\tau} h = 0$, $\bar{\sigma} = 1$, $\alpha = 0.2$.

The trajectories of X and Y positions of the ego-vehicle for risk-averse (i.e., $\alpha > 0$) case is displayed in Fig. 1. Fig. 1 represents two different time segments of the simulation, $[0sec \sim 4sec]$ and $[4sec \sim 12sec]$. The shaded green area and dashed green line represent the results of the low-level continuous-time trajectories based on the high-level actions towards the terminal goal between $[10\%, 90\%]$ quantiles and the means across ten independent experiments. The successive frames of the ego vehicle are shown by the

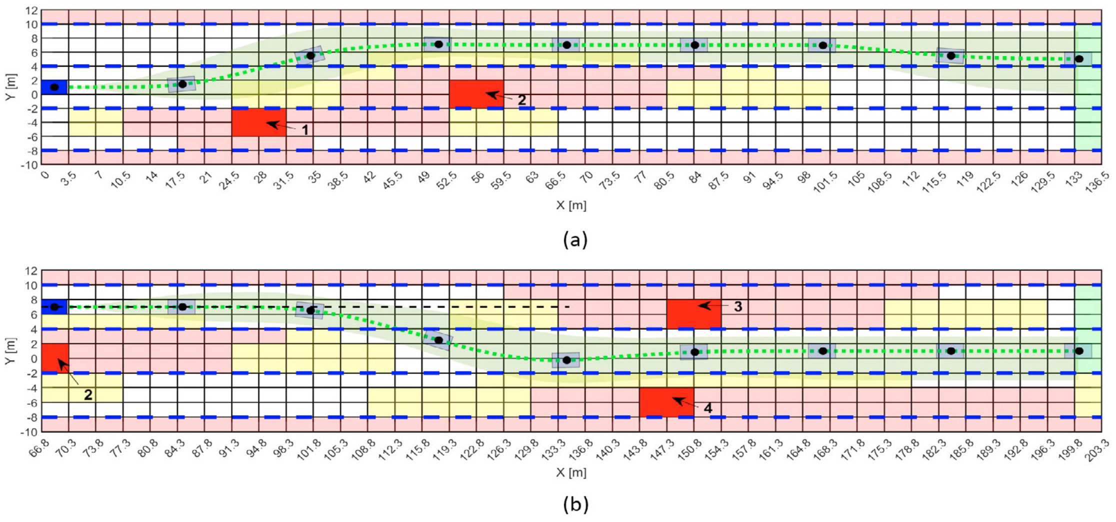

Fig. 1. Three-lane highway represented by 11×39 grid. Obstacles are red. Ego-vehicle is blue. Lanes are shown by dashed blue lines. The shaded green area and dashed green line represent the results between $[10\%, 90\%]$ quantiles and the means across ten independent experiments.

blue square to represent the current position of the ego vehicle and the transparent cyan squares to denote some of the planned estimated positions. Moreover, the current positions of the other vehicles are shown with red squares, and high risky zones and low risky zones are shown with transparent red and yellow squares, respectively, in the grids. Note that at the time $4sec$, the FCU module detects the infeasibility issue and initiates new planning through the high-level risk-averse Q- learning.

5. CONCLUSION

A risk-averse planner is developed for navigation of autonomous vehicles between lanes around moving vehicles. The multi-lane road ahead of a vehicle is represented by a finite-state MDP and the entire planning horizon is modeled as a series of MDPs that evolve over time. A static convex program-based preview-based Q-learning algorithm is then developed that leverages these probabilistic reward values to learn risk-averse optimal planning strategies. It is shown that, for any chosen confidence and constraint violation probabilities, the presented algorithm only needs to sample a finite number of MDP transitions.

REFERENCES

Bäuerle, N. and Jaśkiewicz, A. (2018). Stochastic optimal growth model with risk sensitive preferences. *Journal of Economic Theory*, 173, 181–200.

Bertsekas, D.P. (2018). *Abstract dynamic programming.* Athena Scientific Nashua, NH, USA.

Borkar, V.S. (2002). Q-learning for risk-sensitive control. *Mathematics of operations research*, 27(2), 294–311.

Campi, M.C., Garatti, S., and Prandini, M. (2009). The scenario approach for systems and control design. *Annual Reviews in Control*, 33(2), 149–157.

Dvijotham, K., Fazel, M., and Todorov, E. (2014). Convex risk averse control design. In *53rd IEEE Conference on Decision and Control*, 4020–4025. IEEE.

Geibel, P. and Wysotzki, F. (2005). Risk-sensitive reinforcement learning applied to control under constraints. *Journal of Artificial Intelligence Research*, 24, 81–108.

Hakobyan, A., Kim, G.C., and Yang, I. (2019). Risk-aware motion planning and control using cvar-constrained optimization. *IEEE Robotics and Automation Letters*, 4(4), 3924–3931.

Haskell, W.B. and Jain, R. (2013). Stochastic dominance-constrained markov decision processes. *SIAM Journal on Control and Optimization*, 51(1), 273–303.

Haskell, W.B. and Jain, R. (2015). A convex analytic approach to risk-aware markov decision processes. *SIAM Journal on Control and Optimization*, 53(3), 1569–1598.

Lin, K., Jie, C., and Marcus, S.I. (2018). Probabilistically distorted risk-sensitive infinite-horizon dynamic programming. *Automatica*, 97, 1–6.

Mazouchi, M., Nageshrao, S., and Modares, H. (2021). A risk-averse preview-based q-learning algorithm: Application to highway driving of autonomous vehicles. *arXiv preprint arXiv:2112.03232.*

Mazumdar, E., Ratliff, L.J., Fiez, T., and Sastry, S.S. (2017). Gradient-based inverse risk-sensitive reinforcement learning. In *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, 5796–5801. IEEE.

Meyer, C.D. (2000). *Matrix analysis and applied linear algebra*, volume 71. Siam.

Mihatsch, O. and Neuneier, R. (2002). Risk-sensitive reinforcement learning. *Machine learning*, 49(2), 267–290.

Mueller, M.A. (2017). Reinforcement learning : MDP applied to autonomous navigation.

Ramón Medina, J., Lee, D., and Hirche, S. (2012). Risk-sensitive optimal feedback control for haptic assistance. In *2012 IEEE International Conference on Robotics and Automation*, 1025–1031.

Ratliff, L.J. and Mazumdar, E. (2020). Inverse risk-sensitive reinforcement learning. *IEEE Transactions on Automatic Control*, 65(3), 1256–1263.

Shen, Y., Stannat, W., and Obermayer, K. (2013). Risk-sensitive markov control processes. *SIAM Journal on Control and Optimization*, 51(5), 3652–3672.

Sutton, R.S. and Barto, A.G. (2018). *Reinforcement learning: An introduction.* MIT press.