Offline Reinforcement Learning for Autonomous Driving with Safety and Exploration Enhancement

Tianyu Shi
Department of Civil & Mineral Engineering
University of Toronto
ty.shi@mail.utoronto.ca

Dong Chen
Department of Mechanical Engineering
Michigan State University
chendon9@msu.edu

Kaian Chen
Department of Mechanical Engineering
Michigan State University
chenkaia@msu.edu

Zhaojian Li
Department of Mechanical Engineering
Michigan State University
lizhaoj1@msu.edu

Abstract

Reinforcement learning (RL) is a powerful data-driven control method that has been largely explored in autonomous driving tasks. However, conventional RL approaches learn control policies through trial-and-error interactions with the environment and therefore may cause disastrous consequences such as collisions when testing in real-world traffic. Offline RL has recently emerged as a promising framework to learn effective policies from previously-collected, static datasets without the requirement of active interactions, making it especially appealing for autonomous driving applications. Despite promising, existing offline RL algorithms such as Batch-Constrained deep Q-learning (BCQ) generally lead to rather conservative policies with limited exploration efficiency. To address such issues, this paper presents an enhanced BCQ algorithm by employing a learnable parameter noise scheme in the perturbation model to increase the diversity of observed actions. In addition, a Lyapunov-based safety enhancement strategy is incorporated to constrain the explorable state space within a safe region. Experimental results in highway and parking traffic scenarios show that our approach outperforms the conventional RL method, as well as state-of-the-art offline RL algorithms.

1 Introduction

Autonomous driving has received exceedingly high research interests in the past two decades as it offers the promise of releasing drivers from exhausting driving. While great advances have been achieved in the field of path planning, perception and controls, high-level decision-making remains a challenge especially in mixed traffic with complex and dynamic driving environment. Recently, numerous reinforcement learning (RL) approaches have been applied to autonomous driving tasks and promising results are reported [1–6]. However, conventional RL algorithms evolve through interacting with the environment, via sometimes trial-and-error exploratory actions that make the vehicles vulnerable to accidents in real-world traffic.

Offline RL (also known as batch RL) has been proposed as a promising framework to address the safety issue where agents learn from pre-collected datasets without interacting with the real-world environment. As such, it has received increased interests in safety-critical applications such as decision making in healthcare, robotics, and autonomous driving [7]. In particular, the batch-constrained RL (BCQ) algorithm is proposed in [8], where a state-dependent generative model is used to restrict

Machine Learning for Autonomous Driving Workshop at the 35th Conference on Neural Information Processing Systems (NeurIPS 2021), Sydney, Australia.

predicted actions to be similar to previously observed ones to tackle the issue of extrapolation error caused by erroneously estimating seen state-action pairs. In addition, the authors in [9] exploit the schemes of value penalty factor and policy regularization in the value and policy objective functions to regularize the learned policy towards the expert policy and worthy performance gains on recently proposed offline RL methods are obtained. The aforementioned behavior-constrained approaches essentially restrict the learned policy distribution to resemble the datasets to mitigate the effects of extrapolation error, which on the other hand will generally drive the agents to act conservatively without efficiently exploring the state and action space [8]. This tends to result in poor diversity of seen state-action pairs, which negatively impairs the learning performance.

Learning to explore is an emerging paradigm to address the issue of insufficient exploration [8, 10–12]. For instance, [11] has shown improved exploratory behavior through adding additive Gaussian noise to the parameter vectors on 3 off-policy deep RL algorithms. Deep Deterministic Policy Gradient (DDPG) [10] is then used to independently train an exploration policy by integrating it with an auto-correlated noise added to the actor policy. Despite promising results, the aforementioned approaches apply state-independent noises to enhance exploration, which may not adapt satisfactorily to more diverse environments like the case in autonomous driving.

In this paper, we build upon the state-of-the-art offline RL algorithm, BCQ, and develop a more efficient RL framework with a learnable parameter noise in the perturbation model to enhance exploration and achieve increased diversity in seen actions. Furthermore, Lyapunov-based safety regulation is adopted to enhance the safety in explorations. The main contributions and the technical advancements of this paper are summarized as follows.

1. We build upon BCQ and develop a more efficient and safety-enhanced offline RL framework that are applicable to many safety-critical real-world applications.
2. A novel learnable parameter noise scheme is employed to enhance the diversity of seen actions and a Lyapunov-based risk factor is constructed to restrict the exploratory state space within the safe region.
3. We conduct comprehensive experiments on autonomous driving in both highway and parking traffic scenarios, and the results show that our approach consistently outperforms standard RL and several state-of-the-art offline RL algorithms in terms of driving safety and efficiency.

The remainder of this paper is organized as follows. Section 2 briefly introduces the preliminaries of RL, offline RL and Lyapunov stability theory. The proposed offline RL framework with enhanced safety and exploration efficiency is described in Section 3 whereas experiments, results, and discussions are presented in Section 4. Finally, we conclude the paper and discuss future works in Section 5.

2 Background

2.1 Preliminaries of Reinforcement Learning

In a RL setting, the objective is to learn an optimal policy π^* that maximizes the accumulated return $R = \sum_{t=0}^{T} \gamma^t r_t$, where r_t is the reward at time step t and $\gamma \in (0, 1)$ is the discount factor. More specifically, the agent observes the state $s_t \in \mathcal{S} \subseteq \mathbb{R}^n$ of the environment at each time t, and interacts with the environment by performing an action $a_t \in \mathcal{A} \subseteq \mathbb{R}^m$ according to a policy $\pi(a|s)$. The state-action value function (or Q-function) $Q^\pi(s_t, a_t)$ of a policy π is the expected return when following the policy after taking action a_t in state s_t. The optimal value function $Q^*(s_t, a_t)$, representing the reward of taking action a_t in state s_t followed by the optimal policy π^* through greedy action choices, can be obtained from the following Bellman equation:

$$\mathcal{T}Q^*(s_t, a_t) = E[r(s_t, a_t) + \gamma \sum_{s_{t+1}} \mathcal{P}(s_{t+1}|s_t, a_t) \max_{a_{t+1}} Q^*(s_{t+1}, a_{t+1})], \tag{1}$$

where $\mathcal{T}$ denotes the Bellman operator and $\mathcal{P}(s_{t+1}|s_t, a_t)$ is the transition probability. Off-policy algorithms like Q-learning [13, 14] fit the Q-function with a parametric model Q_θ and update the parameters with sampled data from the experience buffer dataset $\mathcal{D} = (s_t, a_t, r_t, s_{t+1})$ [15]. Actor-critic networks like DDPG [10] adopt two networks: an actor network $\pi_\theta(s)$ for policy learning and

a critic network $Q_\phi(s, a)$ to reduce variance, where the policy network is updated as:

$$\phi \leftarrow \underset{\phi}{\operatorname{argmax}}\, E_{s\in\mathcal{D}}[Q_\theta(s, \pi_\phi(s))]. \tag{2}$$

2.2 Offline Reinforcement Learning

Offline Reinforcement learning is essentially a type of off-policy RL that works on a pre-collected and static dataset $\mathcal{B}$ without the requirement of continuous interactions with the environment [7, 16]. Typically, the dataset $\mathcal{B} = \{(s_t, a_t, r_t, s_{t+1})\}_{t=0:N}$ of unknown quality is first obtained. Batch-Constrained deep Q-learning (BCQ) [8] is a state-of-the-art offline RL method aiming at enforcing the learned policy to be similar to the behavior policy exhibited in the data. BCQ aims to solve a key challenge in offline RL that the values of the seen state-action pairs are often erroneously estimated (also known the as extrapolation error phenomenon). Towards that end, BCQ samples multi-step actions from a generative model (i.e., VAE [17]), which is then used to train the policy by producing actions similar to the ones in the observed data batch:

$$\pi(s) = \underset{\hat{a_i}}{\operatorname{argmax}}\, Q_\theta(s, \hat{a_i}), \tag{3}$$

where $\hat{a}_i = a_i^n + \xi_\phi(s, a_i^n)$ with a_i^n being the action generated from a generative model and $\xi_\phi(s, a_i^n)$ being a perturbation model added to increase the diversity of seen actions [8]. The perturbation model ξ_ϕ is updated as:

$$\phi \leftarrow \underset{\phi}{\operatorname{argmax}} \sum_{(s,a)\in\mathcal{B}} Q_\phi(s, a), \tag{4}$$

and the critic network Q_θ is updated as:

$$\theta \leftarrow \underset{\theta}{\operatorname{argmin}} \sum_{(s,a)\in\mathcal{B}} (y - Q_\theta(s, a)), \tag{5}$$

where y is a combination of the two target Q-values, $Q_{\theta'_1}$ and $Q_{\theta'_2}$, from the target networks and is defined as:

$$y = r + \gamma \max_{\hat{a_i}}[\lambda \min_{j=1,2} Q_{\theta_j{}'}(s', \hat{a_i}) + (1-\lambda) \max_{j=1,2} Q_{\theta'_j}(s', \hat{a_i})], \tag{6}$$

where λ is a parameter that controls the uncertainty introduced from future time steps.

2.3 Lyapunov Stability

Consider the following dynamical system:

$$\frac{dx}{dt} = f(x, u) \equiv f_u(x), \tag{7}$$

where $x(t) \in D \subseteq \mathbb{R}^n$ is the state vector with D being the domain, and $u(t) \in \mathbb{R}^m$ is the control input vector. The closed-loop system is stable at the origin if for any $\epsilon \in R^+$, there exists $\delta(\epsilon) \in R^+$, such that if $||x(0)|| \leq \delta$ then $||x(t)|| \leq \epsilon$ for all $t \geq 0$. Furthermore, the system is asymptotically stable if it is stable and the state goes to zero asymptotically, i.e., $\lim_{t\to\infty} ||x(t)|| = 0$ for all $||x(0)|| < \delta$ [18].

Lyapunov theory [19] is a well-studied method to characterize the stability conditions. Specifically, if there exists a continuously differentiable function $V : \mathbb{R}^n \to \mathbb{R}^+$ for the closed-loop system $f_u(x)$ such that

$$V(0) = 0, \text{ and}, V(x) > 0, \forall x \in D, x \neq 0, \text{and } \nabla_{f_u} V(x) < 0. \tag{8}$$

Here $\nabla_{f_u} V(x)$ is the Lie derivative and defined as

$$\nabla_{f_u} V(x) = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} \frac{dx_i}{dt} = \sum_{i=1}^{n} \frac{\partial V}{\partial x_i} [f_u]_i(x). \tag{9}$$

3 Methodology

3.1 Learning to Explore

In BCQ, a perturbation model $\xi_\phi(s, a)$ parameterized by ϕ is used to generate a noise signal, which is added to the VAE-generated action a to facilitate exploration and increase the diversity of the seen actions. As reported in [11], injecting parameter noises within traditional RL methods can generally promote the exploration. As such, we extend the BCQ algorithm by adding a learnable parameter noise [20] to the perturbation model $\xi_\phi(s, a)$ as $\xi_{\phi'}(s, a)$. Taking a fully-connected layer $y = wx$ as an example, where $x \in \mathbb{R}^p$ and $y \in \mathbb{R}^q$ are the input and output features, respectively, and $w \in \mathbb{R}^{q\times p}$ is the network parameter. Then the corresponding network with perturbation parameter noise is modified as:

$$y = w'x = (\mu^w + \sigma^w \cdot \epsilon^w)x, \tag{10}$$

where the parameters $\mu^w \in \mathbb{R}^{q\times p}$ and $\sigma^w \in \mathbb{R}^{q\times p}$ are learnable parameters of the perturbation network. Here, $\epsilon^w \in \mathbb{R}^{q\times p}$ are noisy random variables that can be learned through back-propagation. The modified perturbation model is thus updated as:

$$\phi' \leftarrow \underset{\phi'}{\operatorname{argmax}} \sum_{(s,a)\in\mathcal{B}} Q_\theta(s, a_i^n + \xi_{\phi'}(s, a_i^n)), \tag{11}$$

where ϕ' is the parameter of the new perturbation model after incorporating the learnable noise parameters.

3.2 Learning to Provide Safety Guarantee

We consider the case that the operation space is defined and restricted based upon those observed within the static dataset $\mathcal{B}$. We aim at enhancing the BCQ algorithm with guaranteed safety. Towards that end, we perform a joint learning framework to obtain the system dynamics in Eqn. 7 together with its Lyapunov function. This collective learning schemes ensures system stability according to the Lyapunov stability criterion introduced in Section 2.3. Specifically, we define a "nominal" closed-loop system dynamics $\bar{f}_{\psi_1}(\cdot)$ and the corresponding Lyapunov function $V_{\psi_2}(\cdot)$ as two neural networks. From [21], it follows that:

$$f_{\psi_1}(s) = \bar{f}_{\psi_1}(s) - \nabla_{f_{\psi_1}} V_{\psi_2}(s) \frac{\sigma\big(\nabla_{f_{\psi_1}} V_{\psi_2}(s)^\top \bar{f}_{\psi_1}(s) + \alpha V_{\psi_2}(s)\big)}{||\nabla_{f_{\psi_1}} V_{\psi_2}(s)||_2^2}, \tag{12}$$

where the structure of $\bar{f}_{\psi_1}$ can be conveniently chosen as random fully connected network whereas the network for Lyapunov function learning is generally chosen as Input Convex Neural Network (ICNN) [22]. Here α is an assigned parameter, and $\sigma(\cdot)$ is a smoothed ReLU activation with a quadratic region in $[0, l]$:

$$\sigma(x) = \begin{cases} 0, & \text{if } x \le 0 \\ \frac{x^2}{2l}, & \text{if } 0 < x < l \\ x - \frac{l}{2}, & \text{otherwise.} \end{cases} \tag{13}$$

By enforcing that no positive component of $\nabla_{f_{\psi_1}} V_{\psi_2}(s)$ is along the direction of $f_{\psi_1}(s)$, according to the afore-mentioned Lyapunov stability theory, the stability of $f_{\psi_1}(s)$ is guaranteed.

Furthermore, in addition to system stability, we also seek to provide safety guarantees with the optimized solution from the exploration policy. According to Eqn. 8, an extended Lyapunov function design can be formulated as the following mini-max based cost function [18]:

$$\inf_{\psi_1} \sup_{s\in\mathcal{B}} \left[\max(0, -V_{\psi_2}(s)) + \max(0, \nabla_{f_{\psi_1}} V_{\psi_2}(s)) + V_{\psi_2}^2(0))\right]. \tag{14}$$

Note that even the convexity of ICNN ensures that $V(\cdot)$ has only a single global optimum [22], it does not require the optimum is at $s = 0$. To address this issue while avoiding increased computational burden and maintaining the function convexity, we perform an internal kernel function shifting [21] to achieve $V(0) = 0$. In the meantime, a small positive term is added to ensure strict positive-definiteness:

$$V_{\psi_2}(s) = \sigma(g(s) - g(0)) + \epsilon||s||^2, \tag{15}$$

where ϵ is a small constant and $g(\cdot)$ is an ICNN. In practice, Eqn. 14 can be solved as the following empirical Lyapunov risk index through Monte Carlo estimation,

$$L_s = E_{s\sim\rho(\mathcal{B})}[\max(0, -V_{\psi_2}(s)) + \max(0, \nabla_{f_{\psi_1}} V_{\psi_2}(s)) + V^2_{\psi_2}(0))], \tag{16}$$

where s is the state variable sampled according to distribution ρ from the data batch $\mathcal{B}$. Finally, the following Lyapunov risk is added to the critic network as:

$$\theta, \psi_1, \psi_2 \leftarrow \underset{\theta,\psi}{\text{argmin}}(\sum_{(s,a)\in\mathcal{B}} (y - Q_\theta(s,a)) + L_s). \tag{17}$$

Pseudo-code of the proposed offline RL algorithm with enhanced safety and promoted exploration is summarized in Algorithm 1, and the major changes from the BCQ algorithm are highlighted in blue.

Algorithm 1 Improved BCQ with safety and exploration enhancement

Input: Batch of data $\mathcal{B}$, horizon T, target network update rate τ, mini-batch size N, number of sampled action n, minimum weighting λ.
Initialize Q-networks Q_{θ_1} and Q_{θ_2}, noisy perturbation network $\epsilon_{\phi'}$, VAE $G_w = (E_{w_1}, D_{w_2})$ and Lyapunov function V_ψ, with random parameters $\theta_1, \theta_2, \phi, w$ and target network $Q_{\theta'_1}$ and $Q_{\theta'_2}$ with $\theta'_1 \leftarrow \theta_1$, $\theta'_2 \leftarrow \theta_2$.
for episode $t = 1$ to T do **do**
  Sample mini-batch of N transitions (s, a, r, s') from $\mathcal{B}$
  $\mu, \sigma = E_{w1}(s, a)$, $\dot{a} = D_{w_2}(s, z)$, $z \sim \mathcal{N}(\mu, \sigma)$
  w $\leftarrow \text{argmin}_w \sum(a - \dot{a})^2 + D_{KL}(\mathcal{N}(\mu, \sigma)||\mathcal{N}(0, 1))$
  Sample n actions: $a_i \sim G_w(s')^n_{i=1}$
  (Explore efficiently) Generate perturbed actions: $a_i \sim a_i + \epsilon'_\phi(s')^n_{i=1}$
  (Guarantee Safety) Compute Lyapunov risk L_s according to Eq.16
  Compute value target y (Eqn. 5)
  $\theta, \psi \leftarrow \text{argmin}_{\theta,\psi}(\sum_{(s,a)\in\mathcal{B}}(y - Q_\theta(s,a)) + L_s)$
  $\phi' \leftarrow \text{argmax}_{\phi'} \sum_{(s,a)\in\mathcal{B}} Q_\theta(s, a^n_i + \xi_{\phi'}(s, a^n_i))$
  Update target network: $\theta'_i \leftarrow \theta_i$
end for
* The major changes from the BCQ algorithm are highlighted in blue.

4 Experiments

4.1 Experimental Setup

We apply our new offline RL framework to autonomous driving tasks, where the open-sourced gym-based environment, highway-env simulator[1], is adapted as our simulation platform. In this platform, vehicle trajectories are generated based on the kinematic bicycle model [23], where the vehicles take continuous-valued actions for steering and throttle controls as defined in [24]. To collect data for offline RL training, a DDPG agent over 5,000 time steps is trained and the experience buffer $\mathcal{B}$ is trained. We use the DDPG implementation from the OpenAI baselines[2]. The proposed approach is experimented on the following two traffic scenarios.

4.1.1 Highway scenario

The highway environment is illustrated in Fig. 1a, where autonomous vehicle (AV, blue) intends to navigate as fast as possible without colliding with the human-driven vehicles (HDVs, green). The AV is expected to make lane changes to overtake slow-moving vehicles whenever possible to achieve higher speed. The reward function is defined as:

$$r = \alpha \times \frac{v - v_{min}}{v_{max} - v_{min}} - \beta \times collision, \tag{18}$$

where v, v_{min}, v_{max} are the current, minimum and maximum speed of the ego-vehicle, respectively, and $\alpha = 0.4$ and $\beta = 1$ are two weighting coefficients.

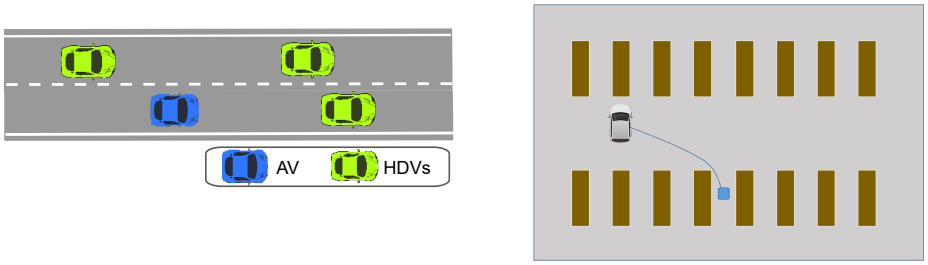
[1] `https://highway-env.readthedocs.io/en/latest/`
[2] `https://stable-baselines.readthedocs.io/en/master/`

4.1.2 Parking scenario

Fig. 1b shows the parking scenario, where the objective of the AV is to park successfully to stay within a desired space with appropriate heading while not colliding with the obstacles (dark green boxes). In this scenario, the reward is defined as:

$$r = -\alpha \times ||s - s_g||^2 - \beta \times violation, \tag{19}$$

where $s = [x, y, v_x, v_y, cos(\phi), sin(\phi)]$ represents the current state of the AV whereas $s_g = [x_g, y_g, 0, 0, cos(\phi_g), sin(\phi_g)]$ is the goal position and orientation. The violation term represents the penalty on hitting obstacles. Here ϕ is the heading angle, and $\alpha = 1$ and $\beta = 5$ are two weighting coefficients.



(a) Lane-change scenario.

(b) Parking scenario.

Figure 1: Two traffic scenarios: freeway and parking.

4.2 Baselines

To demonstrate the effectiveness of our proposed approach, we compare our approach with a state-of-the-art conventional off-policy RL, as well as BCQ, a state-of-the-art offline RL algorithm:

1. *Deep Deterministic Policy Gradient (DDPG)* [10]: DDPG is an off-policy deterministic version of model-free RL algorithm that can handle continuous action space. We adapt the implementation based on OpenAI stable baseline[3].
2. *Batch Constraint Reinforcement Learning (BCQ)* [25]: BCQ is a state-of-the-art offline RL algorithm for continuous control with a state-dependent generative model used to restrict predicted actions to be similar to previous observed ones.
3. *Noisy BCQ*: In this version, we extend BCQ by only adding the exploration-promotion strategy on the policy as detailed in Section 3.1, without employing any safety-enhancement schemes.
4. *Ours*: The framework extends BCQ by incorporating a new perturbation model with learnable parameter noise as well as a Lyapunov-based safety-enhancement scheme.

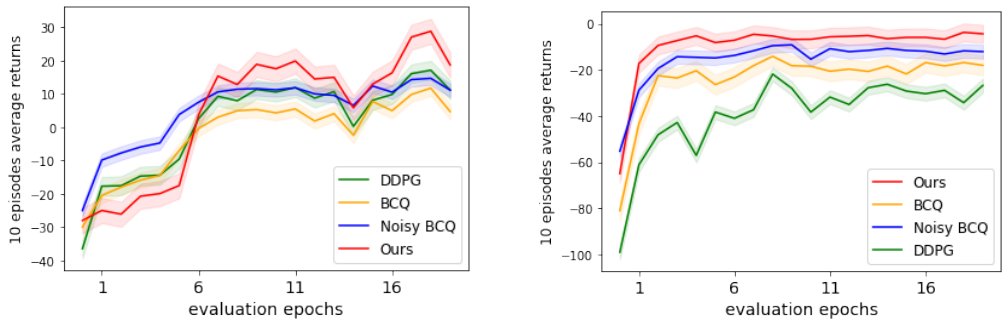
For this comparsion, we train all algorithms over 200 episodes and evaluate the models every 10 episodes with 5 different random seeds while the same random seeds are shared among the models. We set the discount factor γ as 0.7.

4.3 Performance Comparison

4.3.1 Comparison with state-of-the-art benchmarks

The comparison between the proposed algorithm and state-of-the-art off-policy and offline algorithms are shown in Fig. 2a and Fig. 2b on the highway and parking scenarios, respectively. It is clear that our proposed approach consistently outperforms the benchmark algorithms in terms of evaluation returns and training efficiency, which is a result of the proposed parameter noise injection and safety guarantee schemes that facilitate exploration and enhance system safety. It is also noted that Noisy BCQ also outperforms standard BCQ in both traffic scenarios, which demonstrates that adding parameter noises to the perturbation model in BCQ can promote efficient explorations in BCQ.

[3] `https://stable-baselines.readthedocs.io/en/master/`

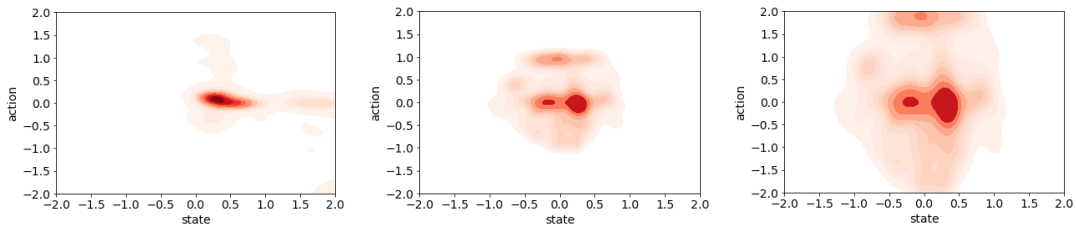


(a) Returns in highway.

(b) Returns in parking.

Figure 2: Comparison on evaluation returns between the proposed approach and state-of-the-art benchmarks.

To show the correlation between state and action pairs, we plot the state-action density in the parking scenario in Fig. 3, where we transform the multi-dimensional features of state and action into one dimensional vectors using principal component analysis (PCA) to show the diversity of the observed state-action pairs. It can be seen that BCQ explores rather "cautiously" with very limited state and action space. In contrast, the Noisy BCQ exhibits more efficient and "aggressive" exploration, surveying a much larger state-action space. This demonstrates that the proposed parameter noise injection scheme can effectively promote exploration in BCQ. With additional Lyapunov-based safety-enhancement, our proposed approach shows the same range of visited action space as Noisy BCQ but restricts the state space in a reasonable range, striking a good balance between exploration and safety as can be seen next.



(a) BCQ

(b) Ours

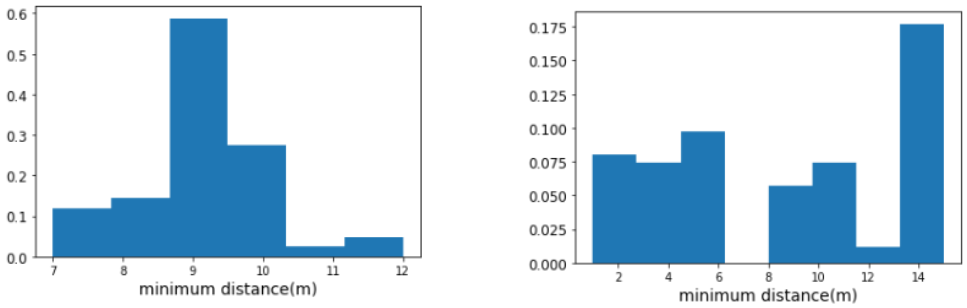
(c) Noisy BCQ

Figure 3: State action density contours in the parking scenario. Darker colors represent more frequent state-action pairs.

4.3.2 Performance of safety enhancement

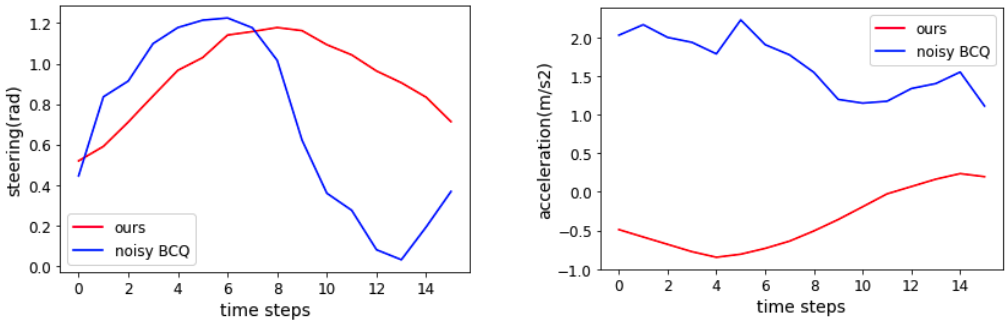
To evaluate the performance of the proposed safety scheme, we compare the proposed approach with the Noisy BCQ that only has the parameter noise injection scheme without safety enhancement. Fig. 4 shows the minimum distance to the surrounding vehicles in the highway scenario for the proposed approach and Noisy BCQ. It is obvious that our approach presents a much higher minimum distance than Noisy BCQ which frequently leads to distances smaller than 5 m. This is because Noisy BCQ only promotes exploration without considering the safety issues.

Furthermore, we compare the performance of our approach with Noisy BCQ in the parking scenario in terms of steering angle, acceleration and success rat. As shown in Fig. 5a, our proposed approach has a smooth steering angle than Noisy BCQ which has sharp changes in steering angle that is risky and leads to poor ride comfort in real-world driving. The acceleration plots in Fig. 5b indicates that our approach also has a lower acceleration compared to Noisy BCQ. Higher and more oscillatory accelerations can cause very poor drive comfort and reduce the lifespan of vehicles. Above all, our approach achieves the highest success rates than the BCQ and Noisy BCQ in the parking scenario as shown in Fig. 6.



(a) Minimum distance with our approach. (b) Minimum distance with noisy BCQ.

Figure 4: Comparison on minimum distance between our method and Noisy BCQ.



(a) Comparison on steering performance. (b) Comparison on acceleration performance.

Figure 5: Performance comparison on steering and acceleration.

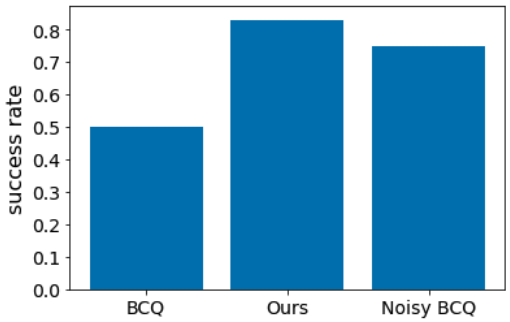


Figure 6: Comparison on different success rates in the parking scenario.

5 Conclusion and Future Work

In this paper, we developed an efficient and safety-enhanced offline RL framework with application to autonomous driving in highway and parking traffic scenarios. To facilitate exploration, we improved the BCQ algorithm by exploiting learnable parameterized noises in the perturbation model. A novel safety scheme was developed using Lyapunov stability theory to enhance safety during explorations. Comprehensive experiments on the application of autonomous driving were conducted to compare our approach with several state-of-the-art algorithms, which demonstrated that the proposed approach consistently outperformed the benchmark approaches in terms of training efficiency and safety. In our future work, we plan to collect and employ more diverse data such as data from conventional control methods and real-world data from autonomous vehicles to further improve the performance.

References

[1] A. E. Sallab, M. Abdou, E. Perot, and S. Yogamani, "Deep reinforcement learning framework for autonomous driving," *Electronic Imaging*, vol. 2017, no. 19, pp. 70–76, 2017.

[2] P. Wang, C.-Y. Chan, and H. Li, "Automated driving maneuvers under interactive environment based on deep reinforcement learning," *arXiv preprint arXiv:1803.09200*, 2018.

[3] T. Shi, P. Wang, X. Cheng, C.-Y. Chan, and D. Huang, "Driving decision and control for autonomous lane change based on deep reinforcement learning," *arXiv preprint arXiv:1904.10171*, 2019.

[4] D. Chen, L. Jiang, Y. Wang, and Z. Li, "Autonomous driving using safe reinforcement learning by incorporating a regret-based human lane-changing decision model," in *2020 American Control Conference (ACC)*. IEEE, 2020, pp. 4355–4361.

[5] J. Chen, S. E. Li, and M. Tomizuka, "Interpretable end-to-end urban autonomous driving with latent deep reinforcement learning," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[6] J. Wu, Z. Huang, C. Huang, Z. Hu, P. Hang, Y. Xing, and C. Lv, "Human-in-the-loop deep reinforcement learning with application to autonomous driving," *arXiv preprint arXiv:2104.07246*, 2021.

[7] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *arXiv preprint arXiv:2005.01643*, 2020.

[8] S. Fujimoto, D. Meger, and D. Precup, "Off-policy deep reinforcement learning without exploration," in *International Conference on Machine Learning*. PMLR, 2019, pp. 2052–2062.

[9] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," *arXiv preprint arXiv:1911.11361*, 2019.

[10] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning," *arXiv preprint arXiv:1509.02971*, 2015.

[11] M. Plappert, R. Houthooft, P. Dhariwal, S. Sidor, R. Y. Chen, X. Chen, T. Asfour, P. Abbeel, and M. Andrychowicz, "Parameter space noise for exploration," *arXiv preprint arXiv:1706.01905*, 2017.

[12] T. Xu, Q. Liu, L. Zhao, and J. Peng, "Learning to explore via meta-policy gradient," in *International Conference on Machine Learning*, 2018, pp. 5463–5472.

[13] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[14] V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, and M. Riedmiller, "Playing atari with deep reinforcement learning," *arXiv preprint arXiv:1312.5602*, 2013.

[15] L.-J. Lin, "Self-improving reactive agents based on reinforcement learning, planning and teaching," *Machine learning*, vol. 8, no. 3-4, pp. 293–321, 1992.

[16] T. Ardoin, E. Paris-Saclay, E. Vinitsky, and A. Bayen, "Extracting traffic smoothing controllers directly from driving data using offline rl."

[17] D. P. Kingma and M. Welling, "Auto-encoding variational bayes," *arXiv preprint arXiv:1312.6114*, 2013.

[18] Y.-C. Chang, N. Roohi, and S. Gao, "Neural lyapunov control," in *Advances in Neural Information Processing Systems*, 2019, pp. 3245–3254.

[19] H. K. Khalil, *Nonlinear System*. Prentice Hall, 2002.

[20] M. Fortunato, M. G. Azar, B. Piot, J. Menick, I. Osband, A. Graves, V. Mnih, R. Munos, D. Hassabis, O. Pietquin *et al.*, "Noisy networks for exploration," *arXiv preprint arXiv:1706.10295*, 2017.

[21] G. Manek and J. Z. Kolter, "Learning stable deep dynamics models," *arXiv preprint arXiv:2001.06116*, 2020.

[22] B. Amos, L. Xu, and J. Z. Kolter, "Input convex neural networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 146–155.

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

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

[25] S. Fujimoto, E. Conti, M. Ghavamzadeh, and J. Pineau, "Benchmarking batch deep reinforcement learning algorithms," *arXiv preprint arXiv:1910.01708*, 2019.