Offline-to-online Reinforcement Learning for Image-based Grasping with Scarce Demonstrations

Bryan Chan[*]
University of Alberta
Edmonton, Canada
bryan.chan@ualberta.ca

Anson Leung & James Bergstra[†]
Ocado Technology
Toronto, Canada

Abstract: Offline-to-online reinforcement learning (O2O RL) aims to obtain a continually improving policy as it interacts with the environment, while ensuring the initial policy behaviour is satisficing. This satisficing behaviour is necessary for robotic manipulation where random exploration can be costly due to catastrophic failures and time. O2O RL is especially compelling when we can only obtain a scarce amount of (potentially suboptimal) demonstrations—a scenario where behavioural cloning is known to suffer from distribution shift. In this work, we propose a novel O2O RL algorithm that can learn in a real-life image-based robotic vacuum grasping task with a small number of demonstrations. The proposed algorithm replaces the target network in off-policy actor-critic algorithms with a regularization technique inspired by neural tangent kernel. We demonstrate empirically that the proposed algorithm exhibits satisficing behaviour after the offline phase and can further reach above 90% success rate in under two hours of interaction time, with only 50 human demonstrations.

Keywords: Image-based grasping, Reinforcement learning, Demonstrations

1 Introduction

Imitation learning (IL) is a popular method for robot learning partly due to the wider data availability, improved data collection techniques, and the development of vision language models [1, 2, 3]. However, while these approaches are more robust to various manipulation tasks, the training requires abundant data. For robotic applications where data is scarce, supervised IL methods such as behavioural cloning are known to suffer from distribution shift [4, 5] and more generally cannot perform better than the demonstrator [6, 7]. Alternatively, we focus on offline-to-online reinforcement learning (O2O RL), which is a two-step algorithm that first pretrains a policy followed by continual improvement with online interactions [8, 9, 10].

As far as we know there is limited success in applying RL on real-life robotic manipulation without using any simulation [11, 12, 13, 14, 15]. One potential reason for this limitation is due to its instability in the learning dynamics. Specifically, most empirically sample-efficient algorithms are off-policy actor-critic algorithms that learn a Q-function [16, 17, 18, 19, 20]—the learned Q-function tends to diverge [21, 22] and overestimate [23]. In deep RL, it has been shown that Q-divergence is correlated to the similarity of the representation between state-action pairs [24, 25]. We hypothesize that **addressing Q-divergence enables sample-efficient RL on real-life robotic manipulation**.

To this end, we describe an O2O RL algorithm that enables training policies on a real-life image-based robotic task in a short amount of time. In particular, we propose to simplify Q-learning by replacing the target network with a regularization term inspired by neural tangent kernel (NTK), which we call Simplified Q. We conduct experiments on a real-life image-based grasping task and

[*]Work done during internship at Ocado Technology.
[†]Work done during employment at Ocado Technology.

compare Simplified Q against behavioural cloning and conservative Q-learning implemented on two commonly-used RL algorithms. We defer the related work and future work to Appendices A and B.

2 Preliminaries and Background

Problem Formulation. A reinforcement learning (RL) problem can be formulated as a Markov decision process (MDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \rho_0, \gamma)$, where $\mathcal{S}$ and $\mathcal{A}$ are respectively the state and action spaces, $r : \mathcal{S} \times \mathcal{A} \rightarrow [0, 1]$ is the reward function, $P \in \Delta^{\mathcal{S}}_{\mathcal{S}\times\mathcal{A}}$ is the transition distribution, $\rho_0 \in \Delta^{\mathcal{S}}$ is the initial state distribution, and $\gamma \in [0, 1)$ is the discount factor. A policy $\pi \in \Delta^{\mathcal{A}}_{\mathcal{S}}$ can interact with the MDP $\mathcal{M}$ through taking actions, yielding random trajectories $\tau = (s_0, a_0, r_0, s_1, a_1, r_1, \cdots)$, where $s_0 \sim \rho_0, a_t \sim \pi(s_t), s_{t+1} \sim P(s_t, a_t), r_t = r(s_0, a_0)$. The return for each trajectory is $G = \sum_{t=0}^{\infty} \gamma^t r_t \leq \frac{1}{1-\gamma}$. We further define the value function and Q-function respectively to be $V^{\pi}_{\gamma}(s) := \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s\right]$ and $Q^{\pi}_{\gamma}(s, a) := \mathbb{E}_{\pi}\left[\sum_{t=0}^{\infty} \gamma^t r_t | s_0 = s, a_0 = a\right]$. The goal is to find a policy π^* that maximizes the expected return for all states $s \in \mathcal{S}$, i.e. $\pi^*(s) = \arg\max_{\pi} V^{\pi}_{\gamma}(s)$.

RL algorithms require exploration which is often prohibitively long and expensive for robotic manipulation. To this end, we consider offline-to-online (O2O) RL, a setting where we are given an offline dataset that is generated by a potentially suboptimal policy. Generally, we can decompose O2O RL into two phases: (1) pretraining an offline agent using offline data, and (2) continually training the resulting agent through extra online interactions. Our goal is to leverage this offline dataset and a limited number of online interactions to train an agent that can successfully complete the task. We consider the setting where we also include offline data during the online interaction [10, 26] and refer $\mathcal{D}$ as the buffer that contains both offline and online data.

Conservative Q-learning (CQL). We build our algorithm on CQL [27], which imposes a pessimistic regularizer on out-of-distribution (OOD) actions to mitigate unrealistically high Q-values on said actions. Suppose the Q-funciton Q_θ is parameterized by θ, the CQL objective is defined as

$$\mathcal{L}_{\text{CQL}}(\theta) := \alpha \left(\mathbb{E}_{\mathcal{D},\mu}\left[Q_\theta(s, a')\right] - \mathbb{E}_{\mathcal{D}}\left[Q_\theta(s, a)\right]\right) + \frac{1}{2}\mathbb{E}_{\mathcal{D}}\left[\left(Q_\theta(s, a) - \mathcal{B}^{\pi}\bar{Q}(s, a)\right)^2\right], \tag{1}$$

where α is a hyperparameter controlling strength of the pessimistic regularizer, μ is an arbitrary policy, $a' \sim \mu(s)$, $a \sim \mathcal{D}$, and $\mathcal{B}^{\pi}\bar{Q}(s, a) := R(s, a) + \gamma\mathbb{E}_{\pi}\left[\bar{Q}(s', a')\right]$ is the empirical Bellman backup operator applied to a delayed target Q-function $\bar{Q}$—when $\bar{Q} = Q$, we use semi-gradient which prevents gradient from flowing through the objective. The first term is the pessimistic Q-value regularization and the second term is the temporal-difference (TD) objective [28]. There can be multiple implementations of CQL. Common implementation chooses $\mu = \pi$ and builds on top of soft actor-critic [29, 17]. Alternatively, crossQ [30] has demonstrated that we can replace the delayed target Q-function with the current Q-function by properly leveraging batch normalization [31]. Calibrated Q-learning (Cal-QL) [9] further augments the regularizer to only penalize OOD actions when their corresponding Q-values exceed the value induced by the dataset $\mathcal{D}$.

3 Stabilizing Q-Learning via Decoupling Latent Representations

Q-learning algorithms are known to diverge [21] and suffer from the overestimation problem [23] even with double-Q learning [25, 32]. We aim to address this problem by proposing a new regularizer. Specifically, our regularizer leverages neural tangent kernel (NTK) to analyze the learning dynamics of Q-learning [25, 24, 22, 33, 34, 35]—the Q-value of a state-action pair $(s', a') \in \mathcal{S} \times \mathcal{A}$ can be influenced by the Q-learning update of another state-action pair $(s, a) \in \mathcal{S} \times \mathcal{A}$. Let θ and θ' be the parameters of the Q-function before and after a stochastic gradient descent (SGD) step respectively, and define $\kappa_\theta(s, a, s', a') = \nabla_\theta Q_\theta(s', a')^\top \nabla_\theta Q_\theta(s, a)$. By performing SGD on the TD objective (second term in Equation (1)) with state-action pair (s, a), we can write the Q-value of another state-action pair (s', a') after the gradient update as

$$Q_{\theta'}(s', a') = Q_\theta(s', a') + \kappa_\theta(s, a, s', a')\left(Q_\theta(s, a) - \mathcal{B}^{\pi}\bar{Q}(s, a)\right) + \mathcal{O}(\|\theta' - \theta\|^2). \tag{2}$$

Here, κ_θ is known as the neural tangent kernel [36] and the last term approaches to zero as the dimensionality increases, a phenomenon known as lazy training [37]. Intuitively, a small magnitude in $\kappa_\theta(s, a, s', a')$ will result in $Q(s', a')$ being less influenced by the update induced by (s, a).

Assuming that the Q-function is parameterized as a neural network $Q_\theta(s, a) := w^\top \Phi(s, a)$, where $\theta = [w, \Phi]$, w is the parameters of the last layer, and $\Phi(s, a)$ is the output of the second-last layer, we can view $\Phi(s, a)$ as a representation layer. Thus, freezing the representation layer during Q-learning update, we can write Equation (2) as

$$Q_{w'}(s', a') = Q_w(s', a') + \Phi(s', a')^\top \Phi(s, a) \left(Q_w(s, a) - \mathcal{B}^\pi \bar{Q}(s, a)\right) + \mathcal{O}(\|w' - w\|^2). \quad (3)$$

Alternative approaches to mitigate this Q-divergence include using target network [38] and double Q-learning [23]. The former is introduced to reduce the influence of rapid changes of nearby state-action pairs during bootstrapping, which can also be addressed by decorrelating their corresponding features. Consequently, we remove the target network, setting $\bar{Q} = Q$, and introduce a regularizer that aims to decouple the representations between different state-action pairs, defined as

$$\mathcal{L}_{\text{reg}}(\Phi) := \mathbb{E}_{s \sim \mathcal{D}, s' \sim \mathcal{D}} \left[\left(\Phi(s, a_u)^\top \Phi(s', a'_\pi) \right)^2 \right], \quad (4)$$

where s, s' are states sampled independently from the buffer $\mathcal{D}$, $\Phi(s, a)$ is the latent representation of state-action pair (s, a), $a_u \sim \mathcal{U}(\mathcal{A})$ is a uniformly sampled action, and $a'_\pi \sim \pi(s')$ is an action sampled from the current policy. Here, we are approximately orthogonalizing the latent representations through minimizing the magnitude of their dot products. This regularizer decorrelates the representations of any two states regardless of the current action, thereby minimizing the influence on other Q-values due to the current update. We build upon conservative Q-learning (CQL) [27] to enable offline training, thus the complete objective for Q-learning is now

$$\mathcal{L}_Q(\theta) := \mathcal{L}_{\text{CQL}}(\theta) + \beta \mathcal{L}_{\text{reg}}(\Phi),$$

where $\beta > 0$ is a coefficient that controls the strength of the decorrelation. We call our method *Simplified Q* as replacing the target network with this regularizer reduces the number of model parameters by half, and further reduces the number of hyperparameters to be a single scalar β, as opposed to having to tune the update frequency and the polyak-averaging term.

4 Experiments

We conduct our experiments on a real-life image-based grasping task. The task consists of controlling a UR10e arm through velocity control to grasp an item inside a bin (Figure 1, left). At each timestep the agent observes a 64×64 RGB image with an egocentric view, proprioceptive information including the pose and the speed, and vacuum pressure reading. This environment has sparse rewards and introduces a challenging exploration problem as we impose minimal boundaries—random policies can easily deviate away from the bin and go further above and outside the bin. Consequently the O2O RL agent must learn to leverage the offline data to accelerate learning. We note that more restricted boundaries may require more engineering (e.g. having precise measurements of the scene) and can prevent the policy from learning optimal behaviours. More details are in Appendix C, and results regarding mitigating Q-divergence and zero-shot generalization are in Appendix D.

Baselines. We compare Simplified Q against behavioural cloning (BC), and CQL built on soft actor-critic (**SAC**) [17] and crossQ (**CrossQ**) [30]. SAC uses a target network to stabilize Q-learning while CrossQ removes the target network with the inclusion of batch normalization [31]. We provide 50 successful human-teleoperated demonstrations and train each policy offline for 100K gradient steps. During the online learning phase, we run each RL algorithm for 200 episodes which corresponds to less than two hours of the total running time—the environment interaction time takes up only 20 minutes while the remaining time is for performing learning updates. For all RL algorithms we enable 3-step Q-learning and symmetric sampling for fairness. The RL agents use a frozen image encoder that is pretrained using successor representation under Hilbert space [39, 40], while the BC agent is trained end-to-end. Extra algorithmic and implementation details are in Appendix E.

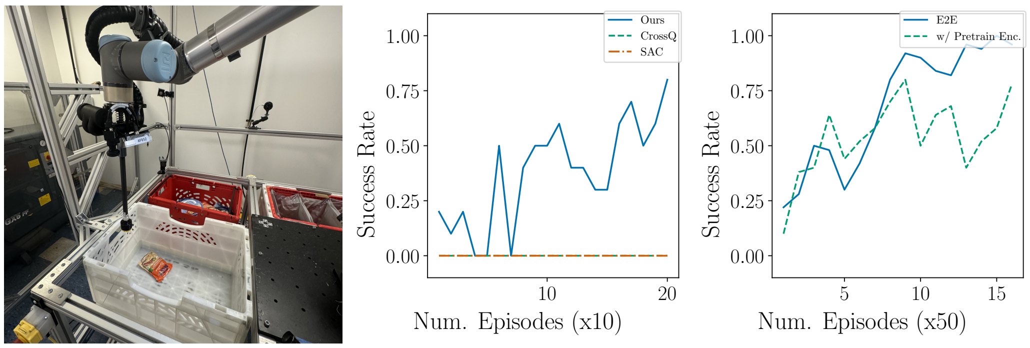


Figure 1: The image-based grasping environment setup and the success rates of various O2O RL algorithms during the online RL phase. All models are first pretrained for 100K gradient steps with 50 human-teleoperated demonstrations. **(Left)** The agent controls a UR10e arm with vacuum suction with the goal of grasping the orange rice bag inside the bin and lifting it well above the bin. **(Middle)** Comparison between SAC, CrossQ, and Simplified Q (Ours). Both SAC and CrossQ have never learned to grasp the item, while Simplified Q can already pick with limited success after offline training. Simplified Q can further achieve up to 75% success rate with only 200 online episodes. **(Right)** Asymptotic performance between training end-to-end (E2E) and frozen pretrained image encoder. The model trained E2E appears to achieve better asymptotic performance than one with a frozen pretrained image encoder.

Results. During the offline phase, BC performs marginally better than simplified Q (Figure 2, top). Behaviourally, both policies reach into the bin 100% of the time, demonstrating satisficing behaviours, whereas CrossQ and SAC fail to learn similar behaviours. We provide visualizations for each agent in Appendix D.1. We then allow the offline-trained policies to further train with extra online interactions. Throughout the 200 episodes both CrossQ and SAC are unable to grasp the item at all, whereas Simplified Q can immediately grasp and finally achieve up to approximately 70% success rate (Figure 1, middle). We also observe that training end-to-end rather than using a pretrained image-encoder can achieve better asymptotic performance, above 90% grasp success rate, without suffering from worse initial performance (Figure 1, right). Finally, comparing Simplified Q with BC, it appears that the BC experiences distribution shift due to the lack of demonstrations. We observe in Figure 2, bottom, that BC requires 500 demonstrations with image-augmentation techniques [41] in order to achieve above 60% success rate. On the other hand, Simplified Q can achieve near 70% success rate within 200 episodes, suggesting O2O RL can perform better than BC by 10% without using more data.

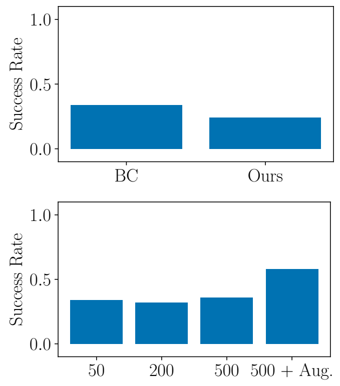


Figure 2: The success rates of offline-trained policies. Each policy is evaluated on 50 attempts. **(Top)** Comparison between BC and Simplified Q (Ours). Simplified Q is able to grasp with limited success while BC performs marginally better than Simplified Q. **(Bottom)** The impact of offline dataset size on BC. Here BC is only able to achieve around 35% success rate until we further include image augmentation from Yarats et al. [41].

5 Conclusion

In this work we introduced a novel regularizer inspired by the neural tangent kernel which can alleviate Q-value divergence. We conducted experiments on a real-life image-based robotic manipulation task, showing that our method could achieve satisficing grasping behaviour immediately after the offline RL phase and further achieve above 90% grasp success rate under two hours of interaction time. We further demonstrated that our method could outperform behavioural cloning and two existing reinforcement learning algorithms with similar amount of total data. These results suggest that we should reconsider whether we truly need large amount of demonstrations for learning near-perfect manipulation policies.

References

[1] A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, et al. Open x-embodiment: Robotic learning datasets and rt-x models. *arXiv preprint arXiv:2310.08864*, 2023.

[2] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn. Learning Fine-Grained Bimanual Manipulation with Low-Cost Hardware. In *Proceedings of Robotics: Science and Systems (RSS)*, July .

[3] S. Haldar, Z. Peng, and L. Pinto. Baku: An efficient transformer for multi-task policy learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[4] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[5] N. Rajaraman, L. Yang, J. Jiao, and K. Ramchandran. Toward the fundamental limits of imitation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 2914–2924, 2020.

[6] T. Xu, Z. Li, and Y. Yu. Error bounds of imitating policies and environments. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 15737–15749, 2020.

[7] A. Ren, S. Veer, and A. Majumdar. Generalization guarantees for imitation learning. In *Conference on Robot Learning (CoRL)*, volume 155, pages 1426–1442, 2021.

[8] Y. Song, Y. Zhou, A. Sekhari, J. A. Bagnell, A. Krishnamurthy, and W. Sun. Hybrid rl: Using both offline and online data can make rl efficient. 2023.

[9] M. Nakamoto, S. Zhai, A. Singh, M. Sobol Mark, Y. Ma, C. Finn, A. Kumar, and S. Levine. Cal-ql: Calibrated offline rl pre-training for efficient online fine-tuning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 62244–62269, 2023.

[10] K. Tan and Z. Xu. A natural extension to online algorithms for hybrid RL with limited coverage. *Reinforcement Learning Journal*, 3:1252–1264, 2024.

[11] J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, and S. Levine. Serl: A software suite for sample-efficient robotic reinforcement learning. *arXiv preprint arXiv:2401.16013*, 2024.

[12] Y. Seo, J. Uruç, and S. James. Continuous control with coarse-to-fine reinforcement learning. In *Conference on Robot Learning (CoRL)*, 2024.

[13] T. Hertweck, M. Riedmiller, M. Bloesch, J. T. Springenberg, N. Siegel, M. Wulfmeier, R. Hafner, and N. Heess. Simple sensor intentions for exploration. *arXiv preprint arXiv:2005.07541*, 2020.

[14] T. Lampe, A. Abdolmaleki, S. Bechtle, S. H. Huang, J. T. Springenberg, M. Bloesch, O. Groth, R. Hafner, T. Hertweck, M. Neunert, et al. Mastering stacking of diverse shapes with large-scale iterative reinforcement learning on real robots. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 7772–7779, 2024.

[15] A. Padalkar, G. Quere, A. Raffin, J. Silvério, and F. Stulp. Guiding real-world reinforcement learning for in-contact manipulation tasks with shared control templates. *Autonomous Robots*, 48(4):12, 2024.

[16] S. Fujimoto, H. Hoof, and D. Meger. Addressing function approximation error in actor-critic methods. In *International Conference on Machine Learning (ICML)*, volume 80, pages 1587–1596, 2018.

[17] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, volume 80, pages 1861–1870, 2018.

[18] T. Hiraoka, T. Imagawa, T. Hashimoto, T. Onishi, and Y. Tsuruoka. Dropout q-functions for doubly efficient reinforcement learning. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[19] X. Chen, C. Wang, Z. Zhou, and K. Ross. Randomized ensembled double q-learning: Learning fast without a model. In *The Ninth International Conference on Learning Representations (ICLR)*, 2021.

[20] T. Ji, Y. Liang, Y. Zeng, Y. Luo, G. Xu, J. Guo, R. Zheng, F. Huang, F. Sun, and H. Xu. Ace: Off-policy actor-critic with causality-aware entropy regularization. In *International Conference on Machine Learning (ICML)*, 2024.

[21] L. Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine learning proceedings 1995*, pages 30–37. Elsevier, 1995.

[22] G. Yang, A. Ajay, and P. Agrawal. Overcoming the spectral bias of neural value approximation. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[23] H. Hasselt. Double q-learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 23, 2010.

[24] A. Kumar, R. Agarwal, T. Ma, A. Courville, G. Tucker, and S. Levine. Dr3: Value-based deep reinforcement learning requires explicit regularization. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[25] Y. Yue, R. Lu, B. Kang, S. Song, and G. Huang. Understanding, predicting and better resolving q-value divergence in offline-rl. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 60247–60277, 2023.

[26] A. Huang, M. Ghavamzadeh, N. Jiang, and M. Petrik. Non-adaptive online finetuning for offline reinforcement learning. *Reinforcement Learning Journal*, 1, 2024.

[27] A. Kumar, A. Zhou, G. Tucker, and S. Levine. Conservative q-learning for offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pages 1179–1191, 2020.

[28] R. S. Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[29] A. Singh, A. Yu, J. Yang, J. Zhang, A. Kumar, and S. Levine. Cog: Connecting new skills to past experience with offline reinforcement learning. *arXiv preprint arXiv:2010.14500*, 2020.

[30] A. Bhatt, D. Palenicek, B. Belousov, M. Argus, A. Amiranashvili, T. Brox, and J. Peters. Crossq: Batch normalization in deep reinforcement learning for greater sample efficiency and simplicity. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[31] S. Ioffe. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[32] T. Ablett, B. Chan, J. H. Wang, and J. Kelly. Value-penalized auxiliary control from examples for learning without rewards or demonstrations. *arXiv preprint arXiv:2407.03311*, 2024.

[33] Q. He, T. Zhou, M. Fang, and S. Maghsudi. Adaptive regularization of representation rank as an implicit constraint of bellman equation. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024.

[34] Y. Ma, H. Tang, D. Li, and Z. Meng. Reining generalization in offline reinforcement learning via representation distinction. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 40773–40785, 2023.

[35] H. Tang and G. Berseth. Improving deep reinforcement learning by reducing the chain effect of value and policy churn. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

[36] A. Jacot, F. Gabriel, and C. Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 31, 2018.

[37] L. Chizat, E. Oyallon, and F. Bach. On lazy training in differentiable programming. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019.

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

[39] T. Moskovitz, S. R. Wilson, and M. Sahani. A first-occupancy representation for reinforcement learning. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[40] S. Park, T. Kreiman, and S. Levine. Foundation policies with hilbert representations. In *International Conference on Machine Learning (ICML)*, 2024.

[41] D. Yarats, I. Kostrikov, and R. Fergus. Image augmentation is all you need: Regularizing deep reinforcement learning from pixels. In *The Ninth International Conference on Learning Representations (ICLR)*, 2021.

[42] H. Zhang, W. Xu, and H. Yu. Policy expansion for bridging offline-to-online reinforcement learning. In *The Tenth International Conference on Learning Representations (ICLR)*, 2022.

[43] J. Li, X. Hu, H. Xu, J. Liu, X. Zhan, and Y.-Q. Zhang. Proto: Iterative policy regularized offline-to-online reinforcement learning. *arXiv preprint arXiv:2305.15669*, 2023.

[44] S. Fujimoto and S. S. Gu. A minimalist approach to offline reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 34, pages 20132–20145, 2021.

[45] P. J. Ball, L. Smith, I. Kostrikov, and S. Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning (ICML)*, volume 202, pages 1577–1594, 2023.

[46] D. Han, B. Mulyana, V. Stankovic, and S. Cheng. A survey on deep reinforcement learning algorithms for robotic manipulation. *Sensors*, 23(7):3762, 2023.

[47] C. Tang, B. Abbatematteo, J. Hu, R. Chandra, R. Martín-Martín, and P. Stone. Deep reinforcement learning for robotics: A survey of real-world successes. *arXiv preprint arXiv:2408.03539*, 2024.

[48] A. Gupta, J. Yu, T. Z. Zhao, V. Kumar, A. Rovinsky, K. Xu, T. Devlin, and S. Levine. Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention. In *2021 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6664–6671, 2021.

[49] A. R. Mahmood, D. Korenkevych, G. Vasan, W. Ma, and J. Bergstra. Benchmarking reinforcement learning algorithms on real-world robots. In *Conference on Robot Learning (CoRL)*, volume 87, pages 561–591, 2018.

[50] Y. Wang, G. Vasan, and A. R. Mahmood. Real-time reinforcement learning for vision-based robotics utilizing local and remote computers. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 9435–9441, 2023.

[51] T. Ablett, B. Chan, and J. Kelly. Learning from guided play: Improving exploration for adversarial imitation learning with simple auxiliary tasks. In *IEEE Robotics and Automation Letters*, volume 8, pages 1263–1270, 2023.

[52] B. Chan, K. Pereida, and J. Bergstra. A statistical guarantee for representation transfer in multitask imitation learning. *arXiv preprint arXiv:2311.01589*, 2023.

[53] J. Garcıa and F. Fernández. A comprehensive survey on safe reinforcement learning. In *Journal of Machine Learning Research*, volume 16, pages 1437–1480, 2015.

[54] T.-Y. Yang, M. Y. Hu, Y. Chow, P. J. Ramadge, and K. Narasimhan. Safe reinforcement learning with natural language constraints. *Advances in Neural Information Processing Systems*, 34:13794–13808, 2021.

[55] P. Pope, C. Zhu, A. Abdelkader, M. Goldblum, and T. Goldstein. The intrinsic dimension of images and its impact on learning. In *The Ninth International Conference on Learning Representations (ICLR)*, 2021.

[56] J. Mei, B. Dai, A. Agarwal, M. Ghavamzadeh, C. Szepesvári, and D. Schuurmans. Ordering-based conditions for global convergence of policy gradient methods. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 36, pages 30738–30749, 2023.

[57] S. Reddy, A. D. Dragan, and S. Levine. Sqil: Imitation learning via reinforcement learning with sparse rewards. In *The Eighth International Conference on Learning Representations (ICLR)*, 2020.

[58] J. Oh, Y. Guo, S. Singh, and H. Lee. Self-imitation learning. In *International Conference on Machine Learning (ICML)*, volume 80, pages 3878–3887, 2018.

[59] D. P. Kingma. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[60] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016.

[61] L. Hussenot, M. Andrychowicz, D. Vincent, R. Dadashi, A. Raichuk, S. Ramos, N. Momchev, S. Girgin, R. Marinier, L. Stafiniak, et al. Hyperparameter selection for imitation learning. In *International Conference on Machine Learning*, pages 4511–4522, 2021.

[62] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning*, pages 1678–1690, 2022.

A Related Work

Reinforcement learning (RL) algorithms require extensive exploration to learn a performant policy, which is an undesirable property for robotic manipulation. Offline-to-online (O2O) RL is an alternative paradigm that also includes policy pretraining with offline data prior to online interactions, with the hope that the pretrained policy is satisficing [8, 10, 26, 42, 43]. Existing algorithms including COG [29], TD3-BC [44], Cal-QL [9], and RLPD [45] have shown some successes in both simulated and real-life environments. While our work draws inspiration from these works, our approach differs in how we learn the Q-function. Our proposed method is also purely RL based, exluding any behavioural-cloning objectives during policy learning.

RL for real-life robotic manipulation tasks often involve sim-to-real transfer [46, 47]. Though, there are several works that directly deploy online RL algorithms on real-life systems [11, 12, 13, 14]. To address the exploration challenges, Hertweck et al. [13] and Lampe et al. [14] leverage hierarchical RL to decompose the task into human-interpretable behaviours. The hierarchical RL agent then explores the space in the temporally-abstracted MDP which enables better state-action coverage. On the other hand, Seo et al. [12] and Luo et al. [11] leverage offline data to provide indirect guidance to the policy. Our work is similar to the latter line of research but also consider the pretraining phase to accelerate online learning.

Finally, our work is most relevant to the recent breakthrough on analyzing Q-learning through the lens of neural tangent kernel (NTK) [36]. Specifically, researchers have identified that the learning dynamics of Q-functions through NTK might describe why Q-learning tends to be unstable and diverge [24, 25, 34, 35]. Consequently the researchers proposed various regularization techniques [33] and model architectures [22] to alleviate this problem. We differentiate ourselves by identifying that the target network in (deep) Q-learning is often applied for similar reasons—decorrelating the consecutive state-action pairs during update. In particular we found that this target network is unnecessary when our regularizer is applied to decorrelate latent representation of state-action pairs.

B Limitations and Future Work

In the future we aim to demonstrate the robustness of Simplified Q through more statistically-sound comparisons and conducting experiments in other manipulation tasks and other domains, particularly longer-horizon tasks. The former is currently limited by the amount of robot resources we have access to—we believe Simplified Q will be quite robust as we have seen positive results on the two experiments we have conducted so far. Secondly, running reinforcement learning algorithms in real life still requires human intervention to reset the environment that may discourage practitioners from applying these algorithms. This limitation might be addressed through reset-free reinforcement learning [48]. Furthermore, the current learning updates are performed in a serial manner, a natural direction is to parallelize this such that the policy execution and policy update are done asynchronously [49, 50].

Our work also takes advantage of using offline data that includes successful attempts to workaround the exploration problem. One question is to investigate whether we can include play data [51] or data collected from other tasks [52]—leveraging multitask data can potentially enable general-purpose manipulation. It is still of interest to perform efficient and safe exploration—for example using vision-language models and/or constrained reinforcement learning algorithms to impose safe policy behaviour and goal generation [53, 54].

There remains a big gap between the theoretical understanding and the empirical results of Simplified Q. Particularly we hope to show that this regularization can bound the Q-estimates to be within realizable returns with high probability, and guarantee convergence of the true Q-function. This may involve analyzing the learning dynamics of temporal difference learning with our regularizer [24, 25]. It will also be interesting to analyze the differences in the learning dynamics with image-based and state-based observations [55]. An alternative theoretical question is to investigate whether Q-overestimation is truly problematic for policy learning [56].

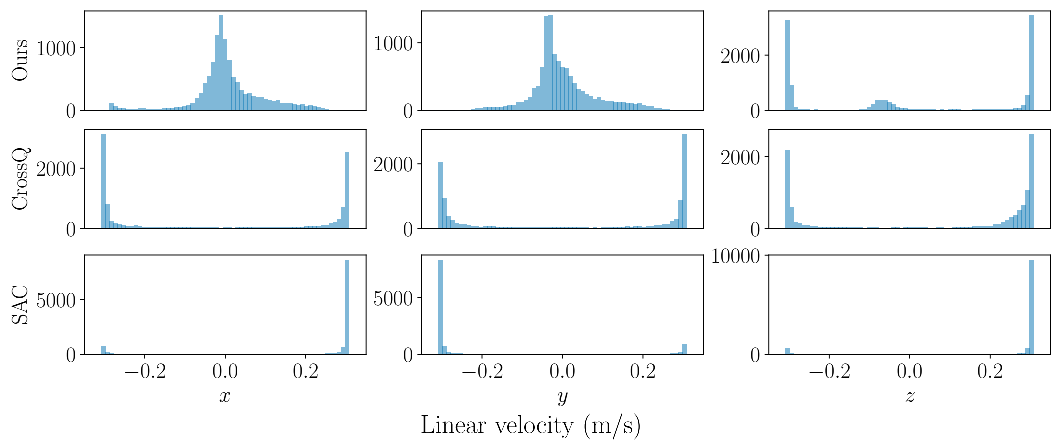


Figure 3: The frequency of actions being taken by the policy during training. We compare Simplified Q (Ours), CrossQ, and SAC. Our policy appears to be able to perform fine-grained actions on the xy axes while CrossQ and SAC exhibits bang-bang behaviours. SAC further appears to have converged into a single direction.

C Environment Details

We conduct our experiments on a real-life image-based grasping task (Figure 1, left). The task consists of controlling a UR10e arm to grasp and lift an item inside a bin. The agent observes a 64×64 RGB image with an egocentric view, the proprioceptive information including the pose and the speed, and the vacuum pressure reading. The agent controls the arm at 10Hz frequency through Cartesian velocity control with vacuum action—a 7-dimensional action space. The agent can attempt a grasp for six seconds. The agent receives a $+1$ reward upon grasping the item and lifting the item above the bin and a $+0$ reward otherwise. In the former, there is a randomized-drop mechanism to randomize the item location, otherwise a human intervenes and changes the item location. The attempts are fixed at six seconds (i.e. episode does not terminate upon success) unless the arm has experienced a protective stop (P-stop)—this enforces the agent to also learn to continually hold the item upon grasping it.

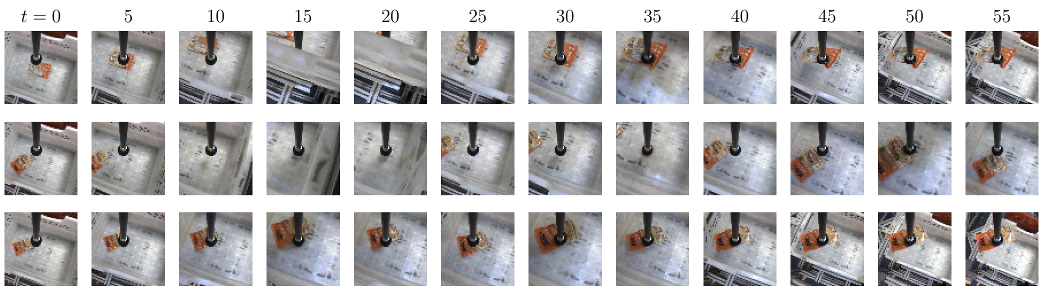
D Extra Experimental Results

D.1 Trajectories over Training

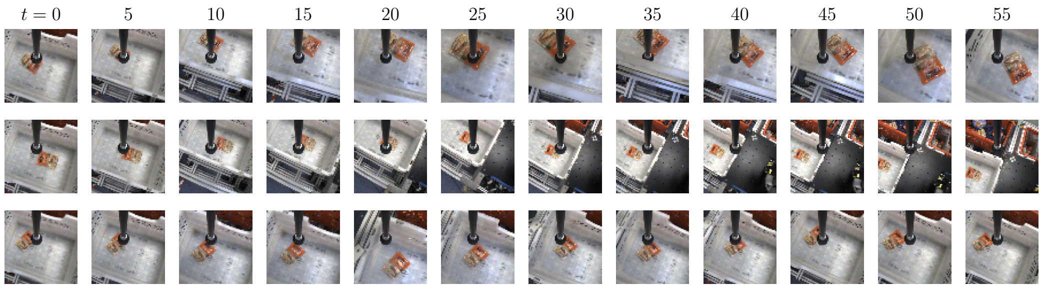
Here we provide few online interaction trajectories for each RL algorithm during training (Figure 4). Behaviourally, the Simplified Q agent can consistently go inside the bin, towards the item, and attempt to pick the item. SAC performs the worst in particular as it has learned to go towards a specific corner in the workspace, away from the bin, which causes the arm to nearly reach singularity. CrossQ hovers around the bin but fails to reach and pick up the item completely. Further visualizing the frequency of an action being taken by each policy during training, SAC has converged to moving towards a single direction while CrossQ has learned to perform bang-bang actions. On the other hand, Simplified Q can perform fine-grained actions on the xy axes which suggests more precise motion (Figure 3).

D.2 Latent Feature Representation Similarity and Q-value Estimates

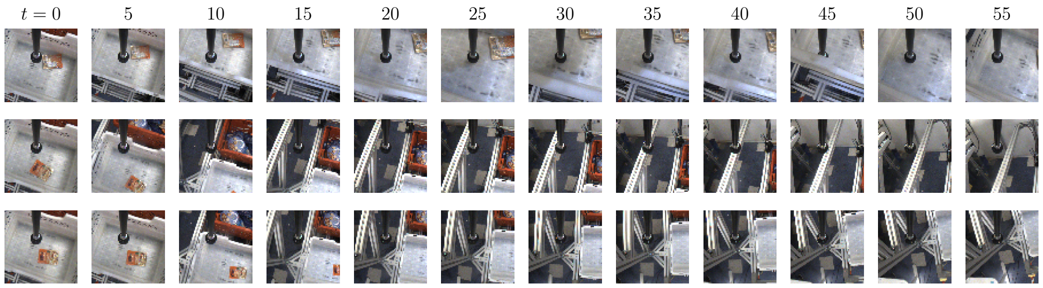
We now analyze the impact of our proposed regularizer. Our goal is to decorrelate the latent feature representation of different state-action pairs, which is measure by the dot product of the features $\Phi(s, a)^\top \Phi(s', a')$, where (s, a) and (s', a') are independent samples from $\mathcal{D}$. We sample 512 random state-action pairs from a buffer of random trajectories and visualize their similarity in the



(a) Ours



(b) CrossQ



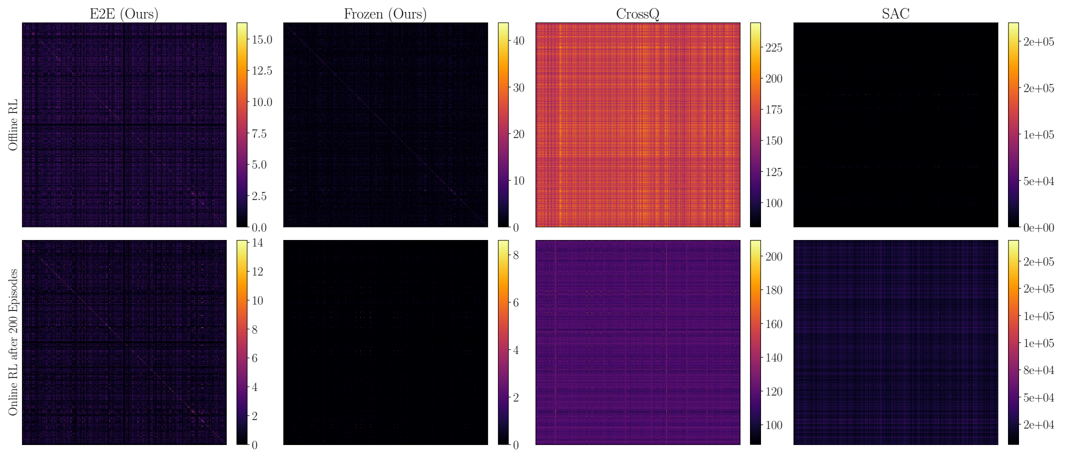
(c) SAC

Figure 4: The n'th online interaction trajectory where $n = \{1, 50, 150\}$, from top row to bottom row, between three RL algorithms. Our algorithm can consistently go inside the bin and attempt to pick the item, while CrossQ and SAC have diverged during training.

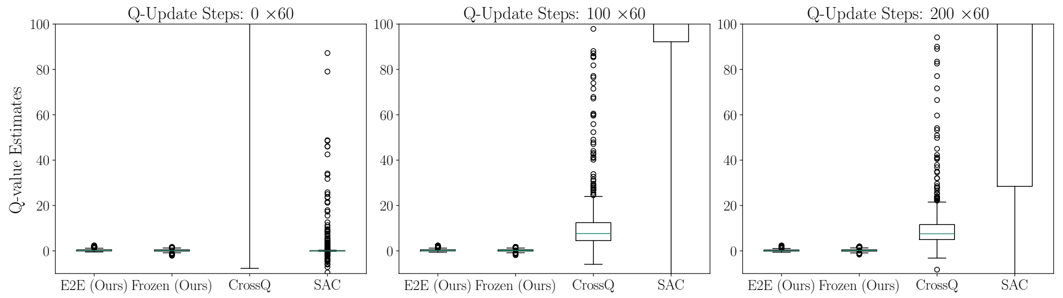
feature space induced by each algorithm during offline RL and online RL. Figure 5a illustrates that using our proposed regularizer yields low-magnitude dot product for most state-action pairs when compared to CrossQ and SAC. We can also observe a general trend that the magnitude decreases for all algorithms as the agent collects more data. We further visualize their Q-values to investigate whether the Q-function suffers from overestimation. We observe in Figure 5b that across 200 online interactions, the Q-values of Simplified Q is consistently below the maximum realizable returns (+60), whereas CrossQ and SAC fail to achieve this. However, there is an interesting trend that SAC appears to begin with reasonable Q-values after the offline phase but quickly diverges during the online phase, while CrossQ has already diverged Q-values after the offline phase. Secondly, we notice another trend that the Q-values are decreasing in magnitude as the agents continually train the Q-function with CrossQ and SAC. We leave the investigation of this phenomenon for future work.

D.3 Zero-shot Generalization

We now investigate the robustness of the RL policy trained with our proposed method without further parameter updates—here we deploy the RL policy trained with 800 online interaction episodes from Figure 1, right. We evaluate the agent on two other item types, one with different colour and one



(a) The similarity between the latent representations, $\Phi(s,a)^\top\Phi(s',a')$, of 512 random state-action pairs after offline RL **(Top)** and 200 episodes of online RL **(Bottom)**. Simplified Q is able to maintain dot-products with small magnitude between different state-action pairs, indicating that the model can decorrelate these state-action pairs, whereas both CrossQ and SAC exhibit significantly larger magnitude dot-products, indicating that these models strongly correlates these different state-action pairs.



(b) The Q-value estimates of 512 random state-action pairs, evaluated at varying number of gradient updates on the Q-function. Simplified Q maintains maintains reasonable Q-values, while both CrossQ and SAC seem to have diverged in Q-values.

Figure 5: (a) The similarity between the latent representation of different state-action pairs. (b) The Q-value estimates of different state-action pairs. From left to right: Model trained end-to-end with Simplified Q (Ours). Model trained with Simplified Q with a frozen pretrained image encoder. Model trained with CrossQ with a frozen pretrained image encoder. Model trained with SAC with a frozen pretrained image encoder.

with different shape and rigidity (Figure 6, right). We also evaluate the agent on other scenarios where there are three items in the bin simultaneously and where the lighting condition changes. We evaluate each scenario for 50 grasp attempts. We emphasize that the agent has only seen a single item over the training runs, making these scenarios totally out-of-distribution (OOD).

Our result is shown in Figure 6, left and we can see that the agent does degrade in performance under OOD scenarios, but we observe that the success rate is around 70% on the different coloured item, while achieving around 50% success rate on the item with different shape and rigidity. The latter fails more frequently as the item is significantly taller, resulting in the agent P-stopping more frequently due to unseen item height. Furthermore, the agent also degrades in performance when there are multiple items in the bin, and we observe that the main faliure mode is when the gripper is in between two items at equidistance, which causes the policy to undercommit on one of the two items. This degradation is more significant when all the items are OOD. Finally, the modified lighting condition does cause a visible performance degradation on the policy, we suspect this is related to the light reflection on the item. Particularly, the policy cannot differentiate between the bottom of the bin and the reflection of the item, thereby neglecting the item completely.

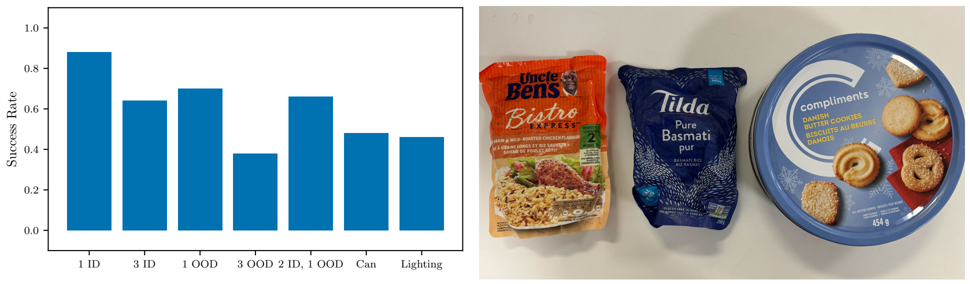


Figure 6: **(Left)** Zero-shot generalization on unseen scenarios using our trained RL policy. We compute the success rate on 50 attempts for each evaluation. The number corresponds to the item count in the bin. *(ID)* In-distribution item: orange rice bag. *(OOD)* Out-of-distribution item: blue rice bag. *(Can)* Out-of-distribution item: cookie can. *(Lighting)* Out-of-distribution lighting condition: Different light source. **(Right)** The item roster for zero-shot evaluation. The trained policy is still able to pick under various OOD scenarios. However including multiple different-coloured items, different item type, and different lighting condition can significant degrade the grasp success rate.

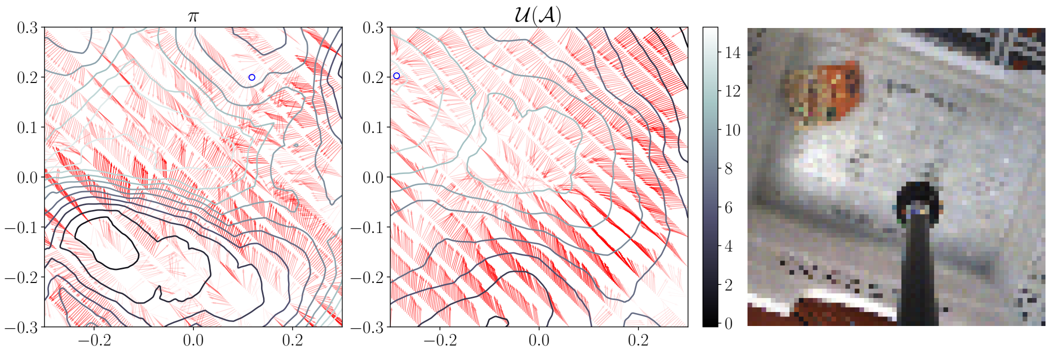


Figure 7: The gradient of Q w.r.t. linear xy velocities and the corresponding image observation. **(Left)** The Q-function is learned through setting $\mu = \pi$. **(Middle)** The Q-function is learned through setting $\mu = \mathcal{U}(\mathcal{A})$. **(Right)** The corresponding image observation. The opacity of the arrow corresponds to the magnitude and the contour corresponds to the Q-values. We observe that with $\mu = \mathcal{U}(\mathcal{A})$ the gradient field tends to point towards the direction of the item whereas $\mu = \pi$ seems to have multiple local optima on different directions.

E Hyperparameters and Algorithmic Details

The self-imitation technique is inspired by soft-Q imitation learning [57] and RLPD [45], where the algorithm samples transitions symmetrically from both the interaction buffer $\mathcal{D}_{\text{on}}$ and the offline buffer $\mathcal{D}_{\text{off}}$. However, symmetric sampling samples half of the transitions from the offline data $\mathcal{D}_{\text{off}}$, which is undesirable when the average return induced by $\mathcal{D}_{\text{off}}$ is lower than the current policy π. We therefore include successful online interaction episodes into $\mathcal{D}_{\text{off}}$ in addition to the interaction buffer $\mathcal{D}_{\text{on}}$, a technique inspired by self-imitation learning [58, 12]. The consequence is twofold: (1) it allows the agent to see more diverse positive examples as it succeeds more, and (2) this results in the buffers being closer to on-policy data as the current policy becomes near optimal.

We choose the CQL regularizer coefficient to be $\alpha = 1.0$ and the NTK regularizer coefficient to be $\beta = 0.2$. We use Adam optimizer [59] with learning rate 0.0003 for offline training and end-to-end online RL; we use a smaller learning rate 0.00005 for online RL with frozen image encoder as we found that using same learning rate is less stable. The batch size is set to be 512 (i.e. sampling 256 from $\mathcal{D}_{\text{off}}$ and 256 from $\mathcal{D}_{\text{on}}$ during online phase), and we perform 60 gradient updates between every attempt for both the policy and Q-functions to avoid jerky motions. The models update at ≥ 1 update-to-data ratio—when P-stop occurs the ratio is higher due to shorter trajectory length.

The image encoder is a ResNet [60], followed by a 2-layer MLP. For the policy, the MLP uses `tanh` activation with 64 hidden units, whereas for the Q-function, the MLP uses `ReLU` activation with 2048 hidden units. Training with a frozen image encoder spends approximately 14 seconds, whereas training end-to-end (E2E) spends approximately 40 seconds. This discrepency comes from updating less parameters for the former—we note that in E2E each model has its own separate image encoder. To pretrain the image encoder, we use first-occupancy Hilbert representation objective introduced by Moskovitz et al. [39] and Park et al. [40]. The image encoder is trained with the images from same 50 demonstrations for 50K gradient steps.

For CQL, we found that setting $\mu = \mathcal{U}(\mathcal{A})$ instead of $\mu = \pi$ to be more effective. Specifically, we visualized the gradient field of the Q-function w.r.t. actions and observe that using $\mu = \pi$ tends to cause the learner policy to get stuck in a local optimum, whereas sampling from $\mathcal{U}(\mathcal{A})$ allows for a smooth landscape with less local optima (Figure 7). Finally, rather than pushing Q-values of in-distribution actions towards infinity in the CQL regularizer, we apply a weighted penalty based on the distance between the in-distribution action and the action sampled from μ. Specifically, the CQL regularizer is implemented as

$$\mathbb{E}_{\mathcal{D},\mu}\left[\left(1 - \exp(-\|a - a'\|)\right) Q_\theta(s, a')\right],$$

where $a \sim \mathcal{D}$ is the in-distribution action and $a' \sim \mu$ is the (possibly) out-of-distribution action.

For BC, we use the exact same policy architecture, learning rate, optimizer, batch size, and number of gradient steps as the RL counterpart, and perform maximum likelihood estimation which corresponds to minimizing the mean-squared error:

$$\mathcal{L}_{BC}(\pi) = \frac{1}{|\mathcal{D}_{\text{off}}|} \sum_{(s,a)\in\mathcal{D}_{\text{off}}} \|a - \pi(s)\|^2.$$

While one may use validation loss to select the "best" policy, we evaluate each checkpoint at every 10K gradient steps and have observed that the performance does not vary significantly, corroborating previous findings [51, 61, 62].