Learning Complicated Manipulation Skills via Deterministic Policy with Limited Demonstrations

Haofeng Liu[1,2], Jiayi Tan[1,2], Yiwen Chen[1,2] and Marcelo H Ang Jr[1,2*]

Abstract

Combined with demonstrations, deep reinforcement learning can efficiently develop policies for manipulators. However, it takes time to collect sufficient high-quality demonstrations in practice. And human demonstrations may be unsuitable for robots. The non-Markovian process and over-reliance on demonstrations are further challenges. For example, we found that RL agents are sensitive to demonstration quality in manipulation tasks and struggle to adapt to demonstrations directly from humans. Thus it is challenging to leverage low-quality and insufficient demonstrations to assist reinforcement learning in training better policies, and sometimes, limited demonstrations even lead to worse performance.

We propose a new algorithm named TD3fG (TD3 learning from a generator) to solve these problems. It forms a smooth transition from learning from experts to learning from experience. This innovation can help agents extract prior knowledge while reducing the detrimental effects of the demonstrations. Our algorithm performs well in Adroit manipulator and MuJoCo tasks with limited demonstrations.

1   INTRODUCTION

While solving individual tasks via manipulators in a controlled setting has led to success in industrial automation, this is less feasible in unstructured settings like the home. Deep Reinforcement Learning (DRL) help manipulator adapt to various environments by interacting with the environment. It has been proven successful in many domains, exceeding human performance in video games [1], robot manipulators[2], and various open-source simulations[3]. However, DRL demands frequent interactions, which proves challenging in manipulator scenarios. Robots may struggle with suboptimal policies and spend time exploring until they accumulate valuable experience, resulting in low sampling efficiency and slow convergence[4].

One approach to addressing these limitations is leveraging human expert demonstrations[5]. In DRL with demonstrations, the agent can learn from both expert behavior and its experience enhancing manipulators' performance in terms of generalization and adaption to unstructured environments[6][7].

Nevertheless, reinforcement learning with demonstrations faces hurdles, such as the demonstration's lack of Markov properties and unrepresentable behaviors. Particularly with limited poor demonstrations, the performance tends to decline[8]. Furthermore, obtaining sufficient samples in practice is often infeasible or prohibitively expensive[4]. Over-reliance on demonstrations may yield unfavorable results[9]. Demonstrations from hand-crafted deterministic policies or rise in human behavior may cause divergence both

---

[*1]Department of Mechanical Engineering, National University of Singapore

[†2]Advanced Robotics Centre, College of Design and Engineering, National University of Singapore

empirically[10] and theoretically [11] [12]. Previous methods like SACfD and DDPGfD[13] expose these issues in our manipulator experiments; thus, we anticipate agents to boost their performance from sub-optimal, even failed data.

We propose TD3 learning from the generator (TD3fG) to solve the above problems. Our methods utilize demonstrations to modify the exploration noise and loss function instead of applying pre-train and experience replies. The results of the comparison experiment exhibit a significant improvement in training efficiency and final performance. To summarize, the main contribution of this work:

• Propose the TD3fG algorithm in section IV. It modifies demonstrations in action noise and loss functions and deploys a smooth transition to avoid overdependence.

• In section VI, we sample 100 demonstrations and show comparison experiments. The results indicate our approach is better adapted to limited demonstrations in complex manipulator tasks.

• In section VII, we perform ablation experiments to investigate the contributions of the individual components in our method.

2   RELATED WORK

2.1   Reinforcement Learning

Researchers have widely applied DRL in auto-driving and robotics and highly investigated it because of its high-level autonomy[14][15]. A renewed interest in RL cascaded from success in bipedal robots and manipulator control [16].

In RL, we assume finishing a task is a Markov Decision Process (MDP) defined by a tuple (S, A, p, r, γ) where S and A are sets of states and actions. $p(s_0|s,a)$ is the state transfer probability. $r : S \times A \to R$ is a reward function and $\gamma \in [0,1)$ is the discount factor. The Markovian property of MDP indicates the conditional probability distribution of the future state only depends on the present state[17].

For the Actor-Critic framework[18], the actor selects actions with the max Q-value to maximize the total reward. The Q value from the critic represents the expectation of future accumulated rewards. It is expressed as $Q_t = \sum_{i=t}^{T} \gamma^{i-t} r_i$.

$$
\begin{aligned}
Q_\pi(s_t, a_t) &= E_{r_t, s_t \sim E, a_i \sim \pi}[Q^\pi | s_t, a_t] \\
&= E_{r_t, s_{t+1} \sim E, a_i \sim \pi}[r_t \\
&+ \gamma E_{a_t \sim \pi}[Q^\pi(s_{t+1}, a_{t+1})]]
\end{aligned}
\tag{1}
$$

2.2   Imitation Learning

Behavior cloning (BC) is one of the simplest forms of imitation learning. It updates the policy network to minimize the mean square error (Deterministic) or cross-entropy (Stochastic) between the output and target actions. For deterministic policy, it is formalized as:

$$
L(\phi_t) = \frac{1}{N} \sum_i (a_{demo} - \pi_{\phi_t}(s_{demo})^2
\tag{2}
$$

BC has achieved great success in driving, locomotion, and navigation but still struggles with limited demonstrations [19]. The bias from experts' track and interference of poor trajectory seriously affects the performance, as observed in several prior works [10][20][21][22]. This weakness becomes obvious in manipulator tasks with high-dimension action and state space.

2

Data Set Aggregation (DAgger) interleaves between expert and learned policy to address the problem of accumulating error [8]. Furthermore, Safe DAgger introduces a safety policy to predict the error from the primary policy[23]. However, DAgger requires access to experts to fix the accumulated errors, which makes it less feasible [24].

2.3 Reinforcement Learning with Demonstrations

Previous works have integrated expert demonstrations to accelerate the training, like DQfD[7], SACfD[25], and DDPGfD[26]. DQfD imports a margin loss which guarantees the expert actions have higher Q-values than other actions. In DDPGfD, the authors import demonstrations into the replay buffer and introduce new methods to incorporate demonstrations.

DQfD, DDPGfD, and SACfD maintain a replay buffer to store demonstrations and experience and draw extra N_D examples from R_D. In further work, researchers introduce a behavior cloning loss only on the demonstration samples[19]. It can omit the pre-training and learn from the demonstration while interacting with the environment. The limitation is sample inefficiency and the need for a large amount of experience, which is impractical for manipulator tasks.

However, the above methods are sensitive to the quantity and quality of the demonstrations. As shown in Section VI, when only providing a small batch of human samples, they cannot get good results as in [25][26][7]

2.4 Twin Delayed Deep Deterministic policy gradient

Twin Delayed Deep Deterministic policy gradient algorithm (TD3) is a deterministic off-policy algorithm that excels in continuous action control[27]. We adopt it as the base algorithm for the RL part and combine it with another approach to learn from demonstrations and control the weight for the demonstration parts.

It alternately interacts with the environment and updates its networks[28]. For every step, the agent executes a selected action from $a_t = \pi_{\phi'}(s_t) + \mathcal{N}_t$ according to policy π and exploration noise $\mathcal{N}$. During the training, the agent samples a random mini-batch N from replay buffer R, then updates the critic according to the Bellman equation and updates the actor to maximize the Q value:

$$y_i = r_i + \gamma min_{i=1,2} Q_{\theta_i'}(s_{i+1}, \pi_{\phi'}(s_{i+1}))$$
$$\theta_i \leftarrow \mathrm{argmin}_{\theta_i} N^{-1} \sum (y - Q_{\theta_i}(s,a))^2 \tag{3}$$
$$\nabla_\phi J(\phi) = N^{-1} \sum \nabla_a Q_{\theta_1}(s,a)|_{a=\pi_\phi(s)} \nabla_\phi \pi_\phi(s)$$

3 PROBLEM FORMULATION

Our problem concerns policy learning for manipulation with limited demonstrations, which has a low average score and is mixed with failed trajectories or human samples that do not fit manipulations well. We focus on tasks that represent a significant fraction of tasks required in our daily life, e.g., tool use and manipulating environmental props[2]. Developing a robust policy with complicated skills needs long-term training. The introduction of demonstrations can facilitate training, but in practical applications, it could be expensive or impractical to collect large-scale, high-quality samples for robots. Our environment comprises dexterous manipulation and interactive objects. We leverage the demonstrations in two ways, committed exploration and imitation, and look into the possible adverse effects of demonstration. By gradually transforming attention from demonstrations to experience, our methods successfully keep efficient training and achieve the best performance in manipulation tasks, as shown in section 6.

3

Figure 1: the flowchart of TD3fG, where the action from generator performs as ground truth in the supervised learning part to calculate the behavior cloning loss L^G. The green trajectory calculates the TD error for the Critic network, and the red and black trajectory gets the weighted error for the actor network.

.

Figure 2: The forward progress of TD3fG, where the generator outputs a combination of reference action from an estimated human policy and a random noise from the OU process. The action noise contains prior knowledge based on demonstrations and could guide the agent's exploration

In this work, we tackle the problem under the setting of reinforcement learning with demonstrations, where the demonstrations are from humans (Adriot) or early-stopped RL agents (MuJoCo) and only 100 trajectories for each task, including suboptimal and failed samples. After achieving noticeable improvement in manipulator tasks, we extend our approach to other robotics tasks (gym MuJoCo)[29] and do ablation studies to analyze the contribution of different components of the method.

4   METHOD

There are two considerations for importing demonstrations, accelerating training and avoiding poor trajectories. We utilize demonstrations in action noise and imitation loss. At early steps imitating the generator. 's actions can help the agent shrink the exploration scope.

4.1 Pre-trained reference action generator

First, the expert demonstrations are used to train a policy neural network G_ϕ through supervised learning. We only prepare 100 trajectories for every task, including sub-optimal and even failed trajectories. section 5.1 shows the max, min, and mean scores of demonstrations.

$$L(\phi) = \frac{1}{N} \sum_i (a_{demo} - G_\phi(s_{demo}))^2 \tag{4}$$

where a_{demo} and s_{demo} are actions and state from the demonstrations.

4.2 Generate exploration noise

The modified exploration noise consists of Ornstein Uhlenbeck noise (from the OU process)[30] and references action noise (from the generator).

The OU noise can improve the exploration efficiency of control tasks in inertial systems and help explore environments with momentum. The noise is presented as $\mathcal{N}$ in $a \leftarrow \pi_\phi(s) + \mathcal{N}$. The differential of OU noise is:

$$dn^o(t) = -\zeta(x_t - \mu)dt + d\sigma W_t$$
$$W_t - W_s \sim \mathcal{N}(0, \sigma^2(t-s)) \tag{5}$$
$$n^o(t) = \mu + (x_0 - \mu)e^{-\zeta t}$$

where μ is mean value, W_t is wiener process. When the state deviates, the noise will pull it close to the mean. The increment in each time interval is the Gaussian distribution. σ^2 represents the variance of a Wiener process, and it determines the magnification of the disturbance.

The second part from the generator is an unstable but meaningful reference action, providing a more explicit exploration direction than random exploration actions. For example, over half of the trajectories in the Hammer fail to drive the nail with the tool, but they provide actions like moving the manipulator close or even picking up the tool. The reference action is added as a bias to the noise to guide committed exploration.

$$a_t \sim \mathcal{N}(a_t, \sigma^2)$$
$$\hat{a}_t \sim \mathcal{N}(a_t + \alpha(t)G(s_t), \sigma^2) \tag{6}$$

The weights are decreased along with training to make agents gradually explore conservatively and exploit experience more. We set $T_1 = T_2 = 0.5T_{max}$ to simplify tuning, where T_{max} is the max training steps.

$$n(t, s_t) = \alpha(t)n^o(t) + \beta(t)G(s_t)$$
$$\alpha(t) = max(1 - \frac{t}{T_1}, 0) \tag{7}$$
$$\beta(t) = max(1 - \frac{t}{T_2}, 0)$$

4.3 Generate BC loss

Third, we import a BC loss between output actions and reference actions. At early steps imitating the actions can help the agent shrink the exploration scope.

$$a_t^{ref} = G(s_t)$$
$$L^G = (a_t^{ref} - \pi_{\phi_t}(s_t))^2 \tag{8}$$

5

L^{BC} has a linear decreasing weight $\varepsilon(t)$ and the policy gradient has an increasing weight $\delta(t)$. It allows the agent to switch from imitation to exploitation gradually.

$$L = \varepsilon(t)L^G(a_t) - \delta(t)Q_{\theta_t}(s_t, a_t)|_{a_t = \pi_{\phi_t}(s_t)}$$
$$\varepsilon(t) = max(1 - \frac{t}{T_{max}}, 0) \tag{9}$$
$$\delta(t) = min(1 - \varepsilon(t), 1)$$

From the aspect of restriction, we expect the agent to select actions with maximum Q value meanwhile stay within a safe distance with the human policy to avoid blind exploration.

$$\pi_{k+1} = \underset{\pi \in \Pi}{\operatorname{argmax}} \, \mathbb{E}_{a \sim \pi(\cdot|x)}[Q(s_t, a_t)],$$
$$s.t. \, ||\pi_k(s_t) - \pi^G(s_t)||^2 < f(t) \tag{10}$$

The safe distance $f(t) = \frac{T_{max}}{T_{max} - t}$ varies from 1 to $+\infty$, where T_{max} is the max training steps, and t is the current number of steps. At the start, $f(t) = 0$, there is a strict restriction to guide exploration, which keeps agent policy close to human policy. In the end, $f(t) = +\infty$ indicates the restriction is completely loosened, and the agent entirely concentrates on maximizing the Q value.

$$\pi_{k+1} = \underset{\pi \in \Pi}{\operatorname{argmin}} \, \mathbb{E}_{a \sim \pi(\cdot|x)}[-Q(s_t, a_t)],$$
$$s.t. \, ||\pi_k(s_t) - \pi^G(s_t)||^2 \frac{(T_{max} - t)}{T_{max}} < 1 \tag{11}$$

Through Lagrange multipliers, we can derive the Lagrange:

$$L(\phi_t, \lambda) = \lambda \varepsilon(t)L^G(a_t) - Q_{theta_t}(s_t, a_t)|_{a_t = \pi_{\phi_t}(s_t)} - \lambda \tag{12}$$

The Lagrangian form is very close to the weighted lost function in eq. (9) except an extra λ. Finding Lagrangian's local minimal is similar to calculating the gradient of the loss function.

5 EXPERIMENT SETUP

We execute the experiments in Adriot manipulation tasks Hammer and Door, which are considered two of the most difficult tasks in this domain. These are typical manipulation tasks with tool use and manipulating environmental props, representing the necessary skills required for our daily robot assistance. Then we expand our tasks to gym MuJoCo tasks, which stands for the robot motion control in a simple environment.

5.1 Environments

We evaluate our methods in Adroit tasks (Hammer and Door), then extend to Mujoco simulations (Ant, Walker2d, and HalfCheetah[29]). To meet our assumptions that the demonstration is limited, we only sample 100 samples with low average scores. The samples for Adriot are from VR equipment that records human behavior. For MuJoCo tasks, the dataset is generated by a partially trained Soft Actor-Critic policy.

(a) Hammer (b) Door

Table 1: Demonstration Details

	Ant	HalfCheetah	Walker2d	Hammer	Door
Average	4635.60	1945.72	2872.05	1096.82	397.83
Max Score	5487.11	4354.80	3914.03	9570.42	1136.83
Min Score	-685.92	-638.49	-5.69	-260.24	-51.03
Std	1356.02	1586.57	1066.74	2286.31	338.91

5.2 Network architecture

The network structure is illustrated in Fig 1. The reference action from the generator affects backpropagation, as shown in Fig 2. In Fig 1, the green arrows calculate the TD error for the Critic network, and the red and black trajectories get the weighted error for the actor network. The red course is the process of exploitation and maximizing Q value. The black course is for imitation based on the distance between the agent's action and the estimated human action. Fig 2 indicates how the agent interacts with the environment. At every step, it executes an action based on the actor's output and the noise. A random OU process noise and a reference action from a pre-trained policy network constitute the noise.

6 EXPERIMENTAL RESULT

6.1 Tasks

Hammer: The manipulator has to use a hammer to drive the nail into the board. The nail is randomly positioned and with significant dry friction.

Door: The manipulator needs to undo the latch and open the door at a random location. The agent should leverage environmental interaction to develop an understanding of the latch.

Ant, Walker2d, and HalfCheetah: Control a two-dimensional bipedal/quadruped robot walk forward.

6.2 Baselines

BC+Finetunes: Directly put the generator in simulation and fine-tune it.

DDPGfD SACfD: They pre-train the model and store the demonstrations in the replay buffer[26].

TD3: Original TD3 without demonstration information.

6.3 Results

Total reward indicates the level of task completion. Fig 3 and Fig 4 compare our work with the TD3, BC + fine-tuning, SACfD, and DDPGfD.

(a) Hammer   (b) Door

Figure 3: Experiment results for Adriot Manipulator tasks, where each curve presents the average total reward of 6 random seeds

Fig 3 gives the results of Adriot Manipulation tasks. Each curve represents the average total rewards with six random seeds. It is clear that TD3fG significantly outperforms other methods. DPGfD and SACfD fail to extract prior knowledge from the given trajectories. A possible explanation is that the samples are directly collected from humans via VR equipment, which may be unsuitable for robots and with low average rewards. Our approach uses demonstrations for exploration rather than directly training policies and shifts attention during training. TD3fG's exceptional performance validates its effectiveness for manipulator control.

(a) Ant   (b) HalfCheetah   (c) Walker2d

Figure 4: Experiment results for MuJoCo tasks, where each curve presents the average total reward of 6 random seeds

To further analyze TD3fG, we extend it to MuJoCo tasks. The results of MuJoCo tasks are in Fig 4. It is hard for TD3 to finish MuJoCo without demonstrations. That may be due to the high dimension of action space and state space. Our method keeps a stable performance and exhibits 2x speeds up and 2x total rewards over the original TD3 in the Ant task.

TD3fG performs best except in HalfCheetah is not as well as SAC. Previous studies suggest that SAC has a distinct advantage in this task[25][31], but our approach worked better with higher training speed in the Ant and Walker2d.

The results reveal that DDPGfD and SACfD depend more on the training samples' quality, as the samples stored in the buffer will act on the entire training process. And their adaptability to different tasks varies considerably. However, our approach is well adapted to different environments and performs particularly well in the manipulator task. Meanwhile, it can leverage a reduced number of low-quality demonstrations and avoid adverse effects, which DDPGfD and SACfD cannot do.

8

Table 2: comparative experimental results

Tasks	hammer	door	Ant	HalfCheetah	Walker2d
Original TD3	2628	592	2375.99	5102.00	2319.36
Behavior Cloning	1861	342	3960.82	5274.89	1943.60
DDPGfD	5283	1248	4194.04	-115.85	4213.81
SACfD	204.5	762.3	2873.46	9241.35	3013.68
TD3fG	9326.71	2345.04	5023.12	7902.64	4065.45

Overall, by introducing a limited number of sub-optimal demonstrations, our approach outperforms other methods in manipulation and various robotics tasks. And it's more lenient on training samples.

7 ABLATION EXPERIMENTS

This section presents ablation experiments to evaluate the influence of different components.

Q-filter + BC loss: The BC loss will be imported if the Q value of reference action from generator $Q(s_t, \pi^{BC}(s_t))$ is bigger than $Q(s_t, \pi(s_t))$.

BC loss + Action noise: This method adds an exploration noise from the pre-trained generator, as in Section IV.

Demonstration Reply: We store 10000 demonstration transitions in the replay buffer.

    (a) Ant   (b) HalfCheetah   (c) Walker2d

Figure 5: Ablation experiment results. From top to bottom are Q-filter, Demonstration Reply, and Action Noise

7.1 Q filter and Decreasing weight

Another way to avoid over-concentrating on demonstrations is the Q-filter. The Q value of the reference action is obtained through the critic, then compared with the Q value of the actor's action. We only import the BC loss if the former is bigger.

$$L^G = 1_{Q(s_t, \pi^G(s_t)) > Q(s_t, \pi_{\phi_t}(s_t))} (a_t^{ref} - \pi_{\phi_t}(s_t))^2 \tag{13}$$

All results with the Q filter are worse, which can be attributed to two possible reasons. First, the Q filter lacks a smooth transition. Second, the critic's misevaluations in the early steps weaken the generator's guidance. The results substantiate that the proposed smooth transition is more helpful with suboptimal and limited demonstrations.

Table 3: Comparison of TD3fG and Q filter

Tasks	Ant	HalfCheetah	Walker2d
TD3fG	4704.44	7601.85	4065.45
Q filter	1809.60	1631.40	3155.78

7.2 Demonstration Reply

Table IV exhibits the comparison results. The total reward lies clearly between DDPGfD and TD3fG. A major difference is a dependence on demonstrations, while TD3fG with demonstration replay is more sensitive to samples' quality than TD3fG. It coincides with our assumption about overreliance on demonstrations.

Table 4: Comparison of Demonstration Reply

Tasks	Ant	HalfCheetah	Walker2d
TD3fG	4704.44	7601.85	4065.45
TD3fG+Replay Buffer	4110.68	3615.32	4047.55

7.3 Action noise

In this part, we add an extra generated Action Noise (AN) and compare it with TD3fG, which only uses BC loss. Table V shows the rewards. Reference action alone boosts the results for all experiments but not as much as TD3fG with BC loss.

8 DISCUSSION AND FUTURE WORK

We propose a new RL with demonstrations algorithm that leverages samples in exploration noise and policy's lost function. TD3fG can extract prior knowledge and avoid poor samples' adverse effects with a smooth transition from learning from demonstrations to learning from experience. Our approach adapts well to

10

Table 5: Comparison of Action Noise

Tasks	Ant	HalfCheetah	Walker2d
TD3fG	4704.44	7601.85	4065.45
AN	1824.43	5527.34	2411.00
TD3fG+AN	4494.95	6913.96	3662.40

limited samples and presents noticeable improvement in manipulator tasks. And it also works well in other robotics tasks. These findings help us to understand the role of unsuitable experts in RL with demonstrations.

A limitation of this study is that the generator's sensitivity to demonstrations remains further investigated. More data collection is required to determine exactly how the quality and quantity of samples affect the final performance and whether our idea can be extended to general robotics tasks.

We will further research how to leverage this kind of demonstration in offline settings, as there may not always be opportunities to interact fully with the environment. The decreasing confidence in demonstrations acts on all expert samples means the agent may ignore some high-quality samples that are worth learning. In future studies, we will train the agent to classify the demonstrations and use different learning methods to handle good trajectories and failure-biased trajectories separately.

9   acknowledgement

References

[1] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. Riedmiller, A. K. Fidjeland, G. Ostrovski, *et al.*, "Human-level control through deep reinforcement learning," *nature*, vol. 518, no. 7540, pp. 529–533, 2015.

[2] M. Andrychowicz, B. Baker, M. Chociej, R. Jozefowicz, B. Mcgrew, J. Pachocki, A. Petron, M. Plappert, G. Powell, and A. Ray, "Learning dexterous in-hand manipulation," 2018.

[3] 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," *Computer ence*, 2015.

[4] J. Wang, H. Guo, Z. Zhu, and Y. Liu, "Policy learning using weak supervision," *Advances in Neural Information Processing Systems*, vol. 34, pp. 19960–19973, 2021.

[5] V. G. Goecks, G. M. Gremillion, V. J. Lawhern, J. Valasek, and N. R. Waytowich, "Integrating behavior cloning and reinforcement learning for improved performance in dense and sparse reward environments," *arXiv preprint arXiv:1910.04281*, 2019.

[6] T. Brys, A. Harutyunyan, H. B. Suay, S. Chernova, M. E. Taylor, and A. Nowé, "Reinforcement learning from demonstration through shaping," in *Twenty-fourth international joint conference on artificial intelligence*, 2015.

[7] F. Codevilla, E. Santana, A. M. López, and A. Gaidon, "Exploring the limitations of behavior cloning for autonomous driving," in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 9329–9338, 2019.

[8] 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*, pp. 627–635, JMLR Workshop and Conference Proceedings, 2011.

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

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

[11] G. Dulac-Arnold, D. Mankowitz, and T. Hester, "Challenges of real-world reinforcement learning," *arXiv preprint arXiv:1904.12901*, 2019.

[12] J. Fu, A. Kumar, O. Nachum, G. Tucker, and S. Levine, "D4rl: Datasets for deep data-driven reinforcement learning," *arXiv preprint arXiv:2004.07219*, 2020.

[13] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging demonstrations for deep reinforcement learning on robotics problems with sparse rewards," *arXiv preprint arXiv:1707.08817*, 2017.

[14] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, *et al.*, "Mastering the game of go with deep neural networks and tree search," *nature*, vol. 529, no. 7587, pp. 484–489, 2016.

[15] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1334–1373, 2016.

[16] S. Gu, E. Holly, T. Lillicrap, and S. Levine, "Deep reinforcement learning for robotic manipulation with asynchronous off-policy updates," in *2017 IEEE international conference on robotics and automation (ICRA)*, pp. 3389–3396, IEEE, 2017.

[17] X. Boyen and D. Koller, "Tractable inference for complex stochastic processes," *arXiv preprint arXiv:1301.7362*, 2013.

[18] M. Andrychowicz, A. Raichuk, P. Stańczyk, M. Orsini, S. Girgin, R. Marinier, L. Hussenot, M. Geist, O. Pietquin, M. Michalski, *et al.*, "What matters for on-policy deep actor-critic methods? a large-scale study," in *International conference on learning representations*, 2021.

[19] A. Nair, B. McGrew, M. Andrychowicz, W. Zaremba, and P. Abbeel, "Overcoming exploration in reinforcement learning with demonstrations," in *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6292–6299, 2018.

[20] A. Kumar, J. Fu, M. Soh, G. Tucker, and S. Levine, "Stabilizing off-policy q-learning via bootstrapping error reduction," *Advances in Neural Information Processing Systems*, vol. 32, 2019.

[21] 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.

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

[23] J. Zhang and K. Cho, "Query-efficient imitation learning for end-to-end autonomous driving," *arXiv preprint arXiv:1605.06450*, 2016.

[24] B. R. Kiran, I. Sobh, V. Talpaert, P. Mannion, A. A. Al Sallab, S. Yogamani, and P. Pérez, "Deep reinforcement learning for autonomous driving: A survey," *IEEE Transactions on Intelligent Transportation Systems*, 2021.

[25] T. Haarnoja, A. Zhou, K. Hartikainen, G. Tucker, S. Ha, J. Tan, V. Kumar, H. Zhu, A. Gupta, P. Abbeel, *et al.*, "Soft actor-critic algorithms and applications," *arXiv preprint arXiv:1812.05905*, 2018.

[26] M. Vecerik, T. Hester, J. Scholz, F. Wang, O. Pietquin, B. Piot, N. Heess, T. Rothörl, T. Lampe, and M. Riedmiller, "Leveraging Demonstrations for Deep Reinforcement Learning on Robotics Problems with Sparse Rewards," *arXiv e-prints*, p. arXiv:1707.08817, July 2017.

[27] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International Conference on Machine Learning*, pp. 1587–1596, PMLR, 2018.

[28] 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.

[29] G. Brockman, V. Cheung, L. Pettersson, J. Schneider, J. Schulman, J. Tang, and W. Zaremba, "Openai gym," 2016.

[30] D. T. Gillespie, "Exact numerical simulation of the ornstein-uhlenbeck process and its integral," *Physical review E*, vol. 54, no. 2, p. 2084, 1996.

[31] 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*, pp. 1861–1870, PMLR, 2018.