Proceeding of the IEEE
International Conference on Robotics and Biomimetics
Dali, China, December 2019

Curiosity-Driven Exploration for Off-Policy Reinforcement Learning Methods*

Boyao Li[1,2], Tao Lu[1], Jiayi Li[1,2], Ning Lu[1,2], Yinghao Cai[1], and Shuo Wang[1,2]
[1]*Research Center on Intelligent Robotic Systems, Institute of Automation, Chinese Academy of Sciences*
[2]*University of Chinese Academy of Sciences*
Beijing, China
{*liboyao2017, tao.lu, lijiayi2019, luning2018, yinghao.cai, shuo.wang*}*@ia.ac.cn*

***Abstract*— Deep reinforcement learning (DRL) has achieved remarkable results in many high-dimensional continuous control tasks. However, the RL agent still explores the environment randomly, resulting in low exploration efficiency and learning performance, especially in robotic manipulation tasks with sparse rewards. To address this problem, in this paper, we introduce a simplified Intrinsic Curiosity Module (S-ICM) into the off-policy RL methods to encourage the agent to pursue novel and surprising states for improving the exploration competence. This method can be combined with an arbitrary off-policy RL algorithm. We evaluate our approach on three challenging robotic manipulation tasks provided by OpenAI Gym. In our experiments, we combined our method with Deep Deterministic Policy Gradient (DDPG) with and without Hindsight Experience Replay (HER). The empirical results show that our proposed method significantly outperforms vanilla RL algorithms both in sample-efficiency and learning performance.**

***Index Terms*— Intrinsic Curiosity Module, Deep Deterministic Policy Gradient, Robotic Manipulation Tasks**

I. Introduction

Recently, deep reinforcement learning (DRL) [1] has made remarkable progresses in learning policies for sequential decision-making problems and applied in various challenging domains from games playing [2] to robotic manipulation learning [3, 4]. However, the common challenge, low sample-efficiency, still remains a major limitation of current RL algorithms for complex tasks with high-dimensional continuous state spaces [5]. Experience replay is one of effective ways to alleviate the sample-inefficiency problems through using a finite-size memory called replay buffer to replay past experiences for policy training. This technique is first introduced in [6] and became popular due to the success of DQN [2] in playing Atari games. Nowadays, many studies have improved this method, such as Prioritized Experience Replay (PER) [7] prioritizes the transitions with high TD-error in the replay buffer more frequently to accelerate learning. Hindsight Experience Replay (HER) [8] replaces the original goals of training samples with that achieved later in the same episodes to encourage the agent to learn much from undesired outcome. Combining Deep Deterministic Policy Gradient (DDPG) method [9] with HER, the agent has the capability to learn more robotic manipulation tasks even in the environments with sparse rewards. Despite these successes, the RL agent still explores the environment randomly, resulting in low exploration efficiency and learning performance, especially in robotic manipulation tasks with high-dimensional continuous state spaces and sparse rewards.

In reinforcement learning, the basic idea of exploration is to encourage the agent to pursue novel or unseen states that visited rarely before to gain more knowledge about its environment. Most existing exploration methods can be classified into two broad categories: count-based exploration and curiosity-driven exploration. Count-based exploration methods aims to use or estimate the visitation counts of states [10] or state-action pairs [11] as an intrinsic bonus to guide the agent to reduce uncertainty, such as Bellemare et al. [12] fits a density generative model to estimate pesudo-count of continuous states and Tang et al. [13] maps states to hash codes to count their occurrences with a hash table. Curiosity-driven exploration method formulates state novelty as the intrinsic reward [14, 15], such as: Intrinsic Curiosity Module (ICM) [16] predicts next states based on current states and actions and sees this prediction error as intrinsic curiosity reward to encourage the agent to discover novel and surprising states beyond its current knowledge about the environment. Episodic Curiosity Module (ECM) [17] compares current observation with the observations in episodic memory to determine the bonus. Most existing curiosity-driven methods can accomplish better performance in environments with high-dimensional continuous state space and even sparse rewards. However, these methods are still restricted to video game environments and combined with on-policy RL algorithms.

In this paper, we introduce a simplified Intrinsic Curiosity Module (S-ICM) into the off-policy RL methods to encourage the agent to pursue novel and surprising states for improving the exploration efficiency. This method can be combined with an arbitrary off-policy RL algorithm. In

*Tao Lu is the corresponding author. This work was supported in part by the National Natural Science Foundation of China under Grant U1713222, 61773378 and partly by the National Key R&D Program of China (grant 2017YFB1300202) and Science and Technology on Space Intelligent Control Laboratory for National Defense, No. KGJZDSYS-2018-09

978-1-7281-6321-5/19//$31.00 ©2019 IEEE

our experiments, we combine our method with DDPG and DDPG+HER respectively and evaluate their performance on three challenging robotic manipulation tasks. Compared with vanilla RL algorithms, S-ICM remarkably improves exploration efficiency and achieves a higher final success rate and faster convergence. Moreover, we demonstrate that our method is compatible with other existing improved methods of experience replay and could further improve their sample efficiency. The policy trained in simulation with our method was deployed on a physical robot and also successfully completed the tasks.

II. Method

In this section, we first introduce the simplified Intrinsic Curiosity Module (S-ICM) and then summarize the detailed training algorithm.

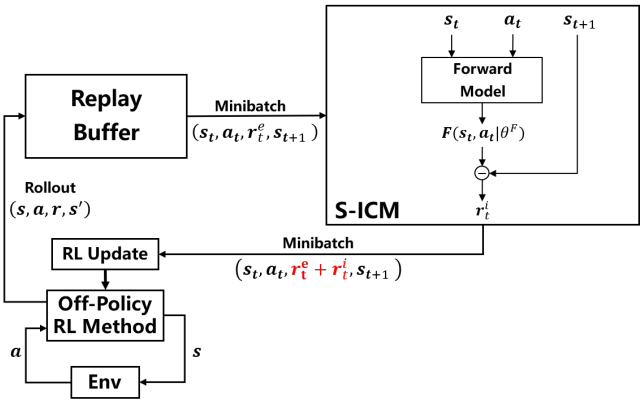


Fig. 1. The detailed architecture of method.

A. Simplified Intrinsic Curiosity Module

We propose a simplified version of the intrinsic curiosity module, called S-ICM, which only utilizes a forward dynamics network on states experienced by the agent and defines the model's state prediction error as an additional intrinsic curiosity reward. The detailed architecture of our method is given in Fig. 1.

We maintain a forward dynamics neural network $F : \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$ parameterized by θ^F to predict the next state given current state and action. When facing novel and unseen states, the prediction error becomes high and the agent will receive high intrinsic rewards. Obviously, pursuing novel states increases the chance of stumbling into the goal states. At each update step, the curiosity module and policy are trained simultaneously using same transitions sampled from the replay buffer, which can speed up convergence as the agent gains more knowledge about its environment. For each training transition, the intrinsic curiosity reward r_t^i is computed as the squared error between the predicted next state $F(s_t, a_t|\theta^F)$ and the actual next state s_{t+1}:

$$r_t^i = clip(\frac{1}{2}\left\|F(s_t, a_t|\theta^F) - s_{t+1}\right\|_2^2, 0, \eta) \tag{1}$$

where $0 < \eta < 1$ is a coefficient that limits the magnitude range of curiosity reward. Instead of using a scaling value [16], we use a *clip* function to keep novelty gap between states to encourage the agent to pursue more novel sates steadily. In this way, we can utilize the intrinsic curiosity rewards to help transforming sparse rewards into dense rewards, which provides more useful information for agent's learning. The policy is trained to optimize the sum total of the extrinsic rewards r_t^e provided by the environment and the intrinsic curiosity rewards r_t^i generated by the forward dynamics neural network. With intrinsic rewards based on the state prediction error of the agent's knowledge about its environment, we encourage agent to visit transitions which are difficult to predict and rare in current replay buffer.

The overall optimization objective is to maximize the expected sum of intrinsic curiosity reward and extrinsic task reward and minimize loss function L_F of the forward dynamics neural network, which can be written as:

$$\min_{\theta^P \theta^F} \left[-\mathbb{E}_{\pi(s_t|\theta^P)}[\sum_t (r_t^e + r_t^i)] + L_F \right] \tag{2}$$

where

$$L_F = \frac{1}{2}\left\|F(s_t, a_t|\theta^F) - s_{t+1}\right\|_2^2 \tag{3}$$

To avoid the agent forgetting the visited paths and exploring those states repeatedly, in this paper, we combine the intrinsic curiosity module with off-policy reinforcement learning algorithm, which enables sampling past experiences from the replay buffer for training and thus ensures exploration stability, especially in large environments with sparse rewards [18].

B. Algorithm

Algorithm 1 shows the complete training details. RL agent interacts with the environment by performing actions using current policy and thus rollouts produced are stored in the replay buffer. At each update, minibatch training transitions are sampled from replay buffer randomly and then passed into the forward dynamics model to obtain respective intrinsic curiosity rewards r_t^i. Combining intrinsic rewards and extrinsic task rewards, RL policy and forward dynamics model are trained simultaneously to encourage agent to seek more novel and surprising states for fast convergence.

III. Experimental Results

In this section, we first introduce the robotic simulation environments we use for the experiments. Second, we combine our method with DDPG and DDPG+HER respectively and compare the performance with vanilla RL algorithms in all three robotic manipulation tasks. Third, we combine

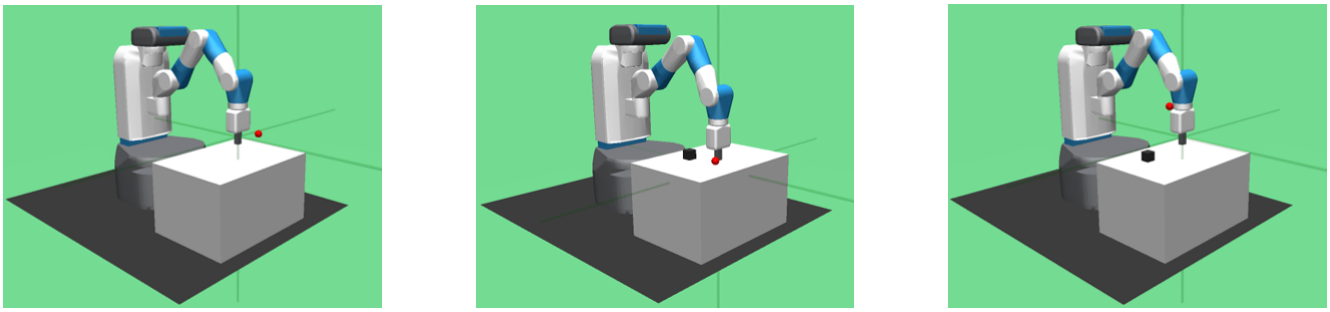


Fig. 2. Robotic simulation environment: FetchReach, FetchPush and FetchPickAndPlace.

Algorithm 1 Simplified Intrinsic Curiosity Module (S-ICM) with off-policy RL method

Given:

- an off-policy RL algorithm $\mathbb{A}$, ▷ e.g. DDPG, DDPG+HER
- a reward function $r : \mathcal{S} \times \mathcal{A} \times \mathcal{G} \rightarrow \mathbb{R}$.

Initialize neural neworks $\mathbb{A}$, forward dynamics model F and replay buffer R
for episode=1, M **do**
  Sample a goal g and an initial state s_0
  for $t = 0, T-1$ **do**
    Sample an action $a_t \leftarrow \pi(s_t \| g)$ using current policy
    Execute the action a_t and observe a new state s_{t+1}
    $r_t^e := r(s_t, a_t, g)$
    Store the transition $(s_t \| g, a_t, r_t^e, s_{t+1} \| g)$ in R
  end for
end for
for batch $= 1, N$ **do**
  Sample a minibatch S from the replay buffer R
  for each transition j in S **do**
    Set r_j^i based on Eq (1) using forward dynamics model F and add it to transition $(s_j \| g, a_j, r_j^e + r_j^i, s_{j+1} \| g)$
  end for
  Train $\mathbb{A}, F$ on S
end for

our method with existing improved method of experience replay, DDPG+HER+PER, to verify its extendibility. Fourth, we check if our method improves sample efficiency in robotic manipulation tasks. Finally, we show the results of the experiments on the physical robot.

A. Simulation Environments

The environment we used in our experiments is a robotic simulation provided by OpenAI Gym [19, 20] which introduces a suite of challenging continuous control tasks based on a 7-DOF Fetch robotic arm with a two-fingered parallel gripper. The robot is simulated using the MuJoCo physics engine [21]. We consider three challenging tasks, including Reach, Push, and Pick&Place, see Fig. 2.

Reach. The task is to move the gripper to a designated goal position.

Push. A block is placed on a table and the task is to move it to a goal position on the table. The robot finger is locked to prevent grasping.

Pick&Place. A block is placed on a table and the task is to grasp it to a goal position which is located either on the table or in the air.

In these robotic manipulation tasks, state is 25-dimensional: includes the positions and linear velocities of the gripper and fingers. If an object is present, it also includes the positions, rotations and angular velocities of object, as well as its position and linear velocities relative to gripper. Action is 4-dimensional: three dimensions specify the desired relative gripper movement and the last dimension controls the opening and closing of fingers. Goal is 3-dimensional that indicates the desired position of end-effector or goal-object (if any). Rewards are sparse and binary: the agent receives a reward of 0 if object is at the target position (within a distance of 5 cm) and -1 otherwise.

B. Performance

We combine our method with two off-policy RL algorithms, DDPG and DDPG+HER, and evaluate their performance on all three robotic manipulation tasks. We use DDPG and DDPG+HER as the baseline respectively and HER utilizes *future* strategy in all experiments for replacing original goals with achieved goals that observed later from the same episodes.

We compare the mean test success rates and the learning curve with respect to training epochs is shown in Fig. 3. Each experiment is carried out across 5 random seeds and the shadow area represents one standard deviation. In all experiments, we use 4 CPUs and train the agent for 25 epochs (each epoch includes 4×100 episodes) in FetchReach environment and 200 epochs in FetchPush and FetchPickAndPlace environments. Throughout the experiments, $\eta = 0.05$ is used for Eq(1) in Reach task, while $\eta = 0.8$ in two remaining

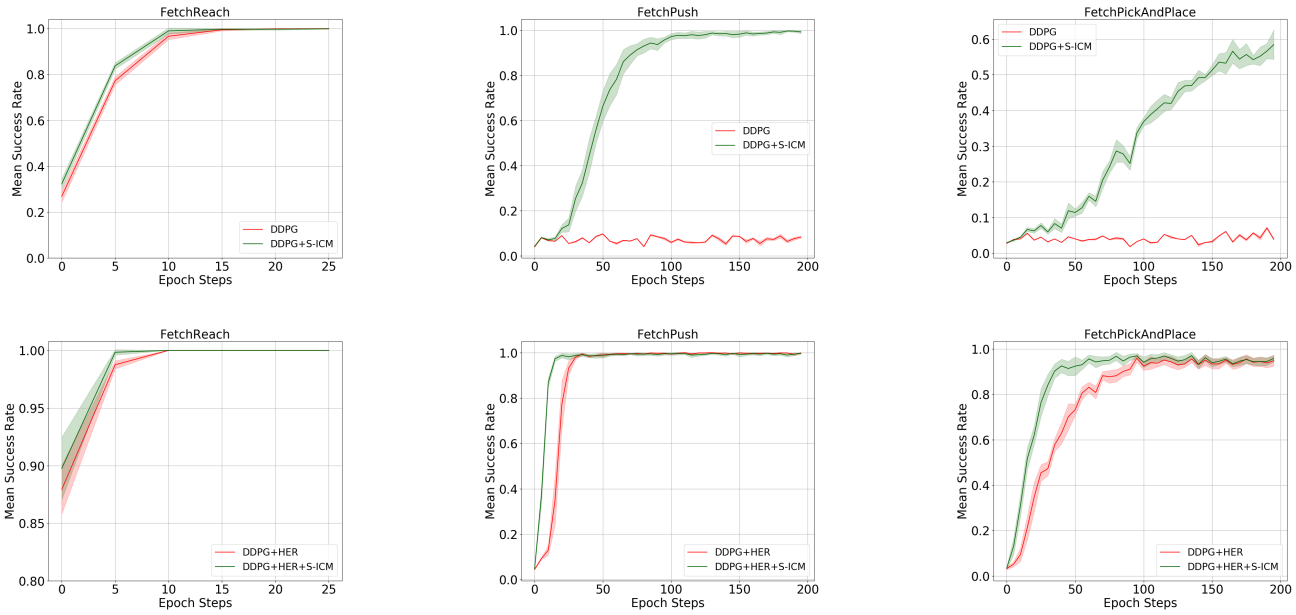


Fig. 3. Comparison on mean test success rate of S-ICM and two baseline methods in all three environments.

tasks. The forward dynamics neutral network we used in S-ICM is a simple framework constructed by concatenating $s_t \| g$ with a_t and passing it into a sequence of two fully-connected layers of 256 and 28 units (the total dimensions of states and goals) respectively. In this way, we hope the forward dynamics model fits the familiar training states but keep insensitive to unseen states, which can help determining state novelty.

From Fig. 3, we can see clearly that S-ICM offers faster convergence and better improvement of final success rate in all three robotic tasks than the baseline methods. Compared with vanilla DDPG, DDPG with S-ICM makes the agent learn faster in Reach task. And the test success rate goes up to 100% and 60% in two other relatively hard tasks while the baseline method even fails learning. Obviously, introducing an exploration strategy to encourage the agent to pursue novel and unseen states increases the chance of stumbling into the goal states and provides more valuable transitions for training. Moreover, compared with DDPG+HER, the proposed method DDPG+HER with S-ICM also accelerates the learning process in all three robotic manipulation tasks. We can see that S-ICM is a simple but effective method to improve exploration efficiency and learning performance of current RL algorithms.

C. Combination with Other Improved Method

The intrinsic curiosity module aims to improve the RL agent's exploration competence by encouraging it to pursue novel and surprising states, but still samples transitions from the replay buffer randomly. And other existing improved

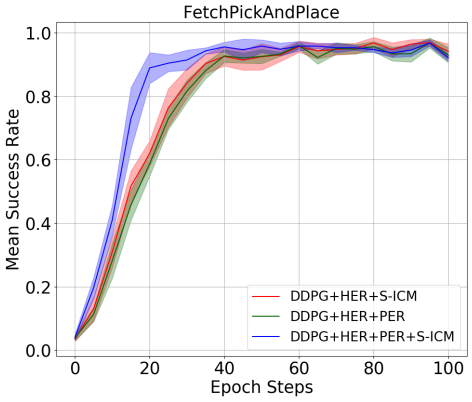


Fig. 4. Comparison of learning curves for three methods on FetchPickAndPlace environment.

method, such as Prioritized Experience Replay (PER), is to prioritize valuable transitions with high TD-errors for replay to learn more efficiently. Therefore, we combine S-ICM with PER to make the best use of replay buffer for improving sample efficiency and agent's learning performance to the most extent. We achieve this objective in relatively harder Pick&Place task, as shown in Fig. 4. We can see from the plot that combining PER+S-ICM with DDPG+HER is a better way to accelerate agent learning and boosting performance compared with using each method separately. Meanwhile, the learning curve also shows high extendibility of S-ICM which can incorporate with other improved methods of experience

replay.

D. Sample Efficiency

To verify if our method improves sample efficiency, we compare the number of episodes needed in the replay buffer when achieving the same mean test success rate using three methods, vanilla DDPG+HER, DDPG+HER with S-ICM and DDPG+HER with PER. From Table I, we can observe that in all three basic robotic simulation environments combining DDPG+HER with S-ICM always needs fewer training samples to reach the same success rate compared with vanilla RL method. Especially in relatively harder Pick&Place task, for the same 97.5% test success rate, DDPG+HER needs 38,200 episodes for training, while DDPG+HER with S-ICM only needs 19,500 episodes, which the sample size has been decreased by nearly one time. Similarly, in the other two simulation environments, S-ICM improves sample-efficiency by factors of 1.84 and 1.69, respectively. Meanwhile, we can also find that our method consumes nearly the same training samples as DDPG+HER+PER, but only 0.6 times of its computational time. In conclusion, S-ICM is able to improve sample-efficiency by an average factor of nearly two (1.83) over the baseline's sample-efficiency.

TABLE I

NUMBER OF EPISODES NEEDED FOR A CERTAIN TEST SUCCESS RATE IN ALL THREE ENVIRONMENTS

	Reach (100%)	Push (100%)	Pick&Place (97.5%)
DDPG+HER	2,240	14,400	38,200
DDPG+HER+PER	**1,100**	10,600	19,600
DDPG+HER+S-ICM	1,220	**8,500**	**19,500**

E. Deployment on a Robotic Arm

We transfer the final learned policy for reach and pick&place tasks trained entirely in the simulation environments with DDPG+HER+S-ICM to real-world UR5 robotic arm respectively. Because the output of policy is the position increment of the end-effector of the robotic arm and it does not care about the robot configuration under the condition of workspace position-reachable, we could directly transfer the policy. The policy execution can be seen as a close-loop control procedure, as shown in Fig. 5. Policy model takes the current observations of real-world environments as input, such as the positions of related objects, goals and the end-effector of robotic arm, et al. and returns the next action to take. The object and goal positions are obtained by color recognition method based on HSV using raw RGB and depth camera images. Considering the policy trained entirely in simulation, it is necessary to first transform the detected observation data from reality to simulation and then transform the output from simulation to reality, using coordinate transformation between virtual and reality [22].

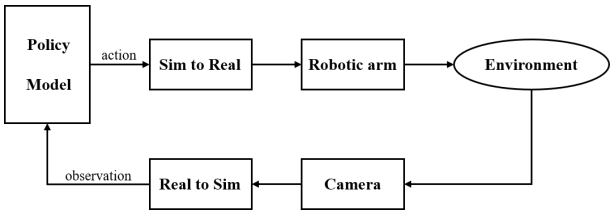
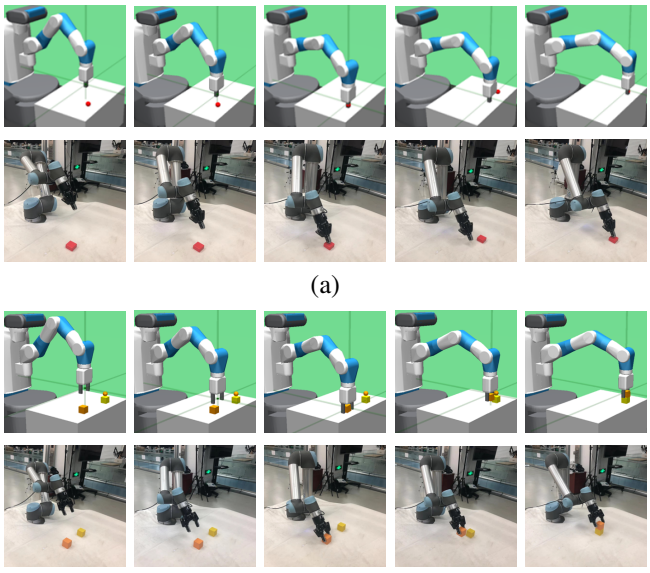


Fig. 5. Flowchart of policy execution.

Fig. 6 shows the process of policy deployed on simulated (top row) and real-world (bottom row) robotic arm. The red block represents the target position in reach task, and the orange block is placed on the yellow block in pick&place task. We test the success rate with 2 cm tolerance error in each task for 20 trials. The policy succeeds in 18 out of 20 trials in reach task, while 15 out of 20 in pick&place task, which shows the consistent performance between simulation environment and real-world scenarios. Failure occurs mainly because of the position estimation error of the blocks caused by the measurement error of RGB-D camera and visual obstruction by the gripper during objects detection.



(a)

(b)

Fig. 6. Frames of learned policy on simulated and real-world robotic arm for (a) reach task and (b) pick&place task.

IV. CONCLUSIONS

In this paper, we combine a simplified Intrinsic Curiosity Module (S-ICM) with off-policy RL algorithm to improve agent's exploration efficiency in robotic manipulation

environments with sparse rewards. Our method shows promising experimental results in all three challenging robotic manipulation tasks and improves both sample efficiency and learning performance compared with vanilla RL methods. Moreover, this method can be combined with other improved methods of experience replay and we demonstrate that with PER. We also show that the policy trained in simulation for reach and pick&place tasks perform well on the physical robot. In future works, except for pursuing states novelty, we aim to go on to develop the task-relevant curiosity-driven exploration method to maximize the sample efficiency.

REFERENCES

[1] R. S. Sutton and A. G. Barto. Reinforcement learning: An introduction. MIT Press, 1998.

[2] V. Mnih, K. Kavukcuoglu, D. Sliver, et al. Human-level control through deep reinforcement learning. Nature, vol. 518, no. 7540, pp. 529-533, 2015.

[3] S. Levine, P. Pastor, A. Krizhevsky, et al. Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection. The International Journal of Robotics Research, vol. 37, no. 4-5, pp. 421-436, 2018.

[4] D. Kalashnikov, A. Irpan, P. Pastor, et al. Qt-opt: Scalable deep reinforcement learning for vision-based robotic manipulation. arXiv preprint arXiv:1806.10293, 2018.

[5] W. Shi, S. Song, H. Wu, et al. Regularized anderson acceleration for off-policy deep reinforcement learning. arXiv preprint arXiv:1909.03245, 2019.

[6] 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.

[7] T. Schaul, J. Quan, I. Antonoglou, et al. Prioritized experience replay. arXiv preprint arXiv 1511.05952, 2015.

[8] M. Andrychowicz, F. Wolski, A. Ray, et al. Hindsight experience replay. Advances in Neural Information Processing Systems, pp. 5048-5058, 2017.

[9] T. P. Lillicrap, J. J. Hunt, A. Pritzel, et al. Continuous control with deep reinforcement learning. arXiv preprint arXiv:1509.02971, 2015.

[10] T. L. Lai, H. Robbins. Asymptotically efficient adaptive allocation rules. Advances in Applied Mathematics, vol. 6, no. 1, pp. 4-22, 1985.

[11] Z. Xu, X. Chen, L. Cao, et al. A study of count-based exploration and bonus for reinforcement learning. IEEE International Conference on Cloud Computing and Big Data Analysis, pp. 425-429, 2017.

[12] M. G. Bellemare, S. Srinivasan, G. Ostrovski, et al. Unifying count-based exploration and intrinsic motivation. Advances in Neural Information Processing Systems, pp. 1471-1479, 2016.

[13] H. Tang, R. Houthooft, D. Foote, et al. #Exploration: A study of count-based exploration for deep reinforcement learning. Advances in Neural Information Processing Systems, pp. 2753-2762, 2017.

[14] Y. Burda, H. Edwards, D. Pathak, et al. Large-scale study of curiosity-driven learning. arXiv preprint arXiv:1808.04355, 2018.

[15] Y. Burda, H. Edwards, A. Storkey, et al. Exploration by random network distillation. arXiv preprint arXiv:1810.12894, 2018.

[16] D. Pathak, P. Agrawal, A. A. Efros, et al. Curiosity-driven exploration by self-supervised prediction. International Conference on Machine Learning, pp. 2778-2787, 2017.

[17] N. Savinov, A. Raichuk, R. Marinier, et al. Episodic curiosity through reachability. arXiv preprint arXiv:1810.02274, 2018.

[18] H K. Yang, P H. Chiang, K W. Ho, et al. Never forget: Balancing exploration and exploitation via learning optical flow. arXiv preprint arXiv:1901.08486, 2019.

[19] M. Plappert, M. Andrychowicz, A. Ray, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. arXiv preprint arXiv:1802.09464, 2018.

[20] G. Brockman, V. Cheung, L. Pettersson, et al. Openai gym. arXiv preprint arXiv:1606.01540, 2016.

[21] E. Todorov, T. Erez, Y. Tassa. Mujoco: A physics engine for model-based control. IEEE/RSJ International Conference on Intelligent Robots and Systems, pp. 5026-5033, 2012.

[22] B. Li, T. Lu, X. Li, et al. An automatic robot skills learning system from robot's real-world demonstrations. Chinese Control And Decision Conference, pp. 5138-5142, 2019.