2024 IEEE International Conference on Robotics and Automation (ICRA 2024)
May 13-17, 2024. Yokohama, Japan

Experience Consistency Distillation Continual Reinforcement Learning for Robotic Manipulation Tasks

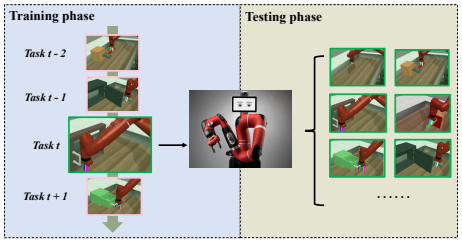
Chao Zhao, Jie Xu, Ru Peng, Xingyu Chen, Kuizhi Mei*, Xuguang Lan*

***Abstract*— Continual reinforcement learning, which aims to help robots acquire skills without catastrophic forgetting, obviating the need to re-learn all tasks from scratch. In order to enable lifelong acquisition of skills in robots, replay-based continual reinforcement learning has emerged as a promising research direction. These techniques replay data from previous tasks to mitigate forgetting when learning new skills. However, existing replay-based methods store poor representative experience, and the experience utilization of old tasks is inefficient. To address these issues, we propose an experience consistency distillation method for robot continual reinforcement learning to improve the data efficiency of the experience. Specifically, the experience of old tasks are distilled to obtain Markov Decision Process (MDP) data with high compression ratio and information content. To ensure consistent data distributions before and after distillation, we further utilize a Fréchet Inception Distance (FID) loss as a regularization constraint. In order to improve experience utilization efficiency, the policy is then trained using both the distilled data and current task data, with policy distillation performed based on uncertainty metrics. Our method is validated in the continual reinforcement learning simulation platform and real scene with a UR5e robot arm. Experimental results indicate that our method achieves higher success and lower buffer size requirement compared to other methods.**

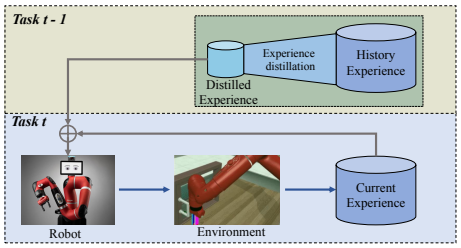
I. Introduction

Continual reinforcement learning aims to enable robotic adaptation in non-stationary environments and progressive skill development, while retaining reasoning in old environments, as shown in Fig. 1(a). Due to real-world complexity and non-stationarity, robots trained by traditional reinforcement learning struggle to adapt to changing environments [1]. For instance, a robot trained for indoor navigation would fail outdoors [2]. And one that learns pushing may lose that skill when learning grasping [3]. Continual reinforcement learning has attracted great interest due to its ability to adapt robots to non-stationary environments. [4], [5].

Catastrophic forgetting, as the major problem in continual reinforcement learning, refers to the inevitable forgetting of old task experiences when learning a new task. Apart from replay-based methods [6], there are common approaches against catastrophic forgetting include regularization-based methods [7], [8] and architecture-based methods [9], [10]. However, the performance of these methods tends to degrade and hit a plateau as the number of tasks increases. Replay-based methods mitigate catastrophic forgetting by storing a small amount of old task experience in a buffer and replaying it with new data during training [11]. The representative strategies include sampling representative samples or maintaining coreset for each task [12]–[14]. However, these methods are fundamentally clustering methods, which have strict requirements on initial cluster centers and can become trapped in local optima. Thus, representation ability of experience stored by these methods is insufficient and suffers from distribution shift.



(a) Continual reinforcement learning for robotics



(b) Experience consistency distillation for continual RL

Fig. 1. (a) The paradigm of continual reinforcement learning. In training phase, the robot learns one task at a time without knowing the next task. In testing phase, the robot needs to maintain inference capabilities on all learned tasks. (b) The distilled experience obtained by our proposed experience consistency distillation can better characterize the old task data, effectively alleviating catastrophic forgetting in continual reinforcement learning.

In recent years, dataset distillation has gained increasing research attention in image recognition [15], [16]. By distilling a large dataset into a small one, training only on the distilled dataset can approach the performance of training on the full dataset, greatly reducing computational and storage costs [17]. Since the distilled dataset has stronger representation ability, we consider using it to obtain replay experience with low redundancy. Current dataset distillation methods mainly include optimizing based on target domain validation gradients [17] and matching training trajectories [16]. However, these methods are designed for supervised

* Corresponding author
C. Zhao, J.Xu, R. Peng, X. Chen, K. Mei and X. Lan are with the Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University, Xi'an 710049, China (e-mail: zhaochaocl@stu.xjtu.edu.cn; jie.xu@stu.xjtu.edu.cn; pengr0925@163.com; xingyuchen1990@gmail.com; meikuizhi@xjtu.edu.cn; xglan@mail.xjtu.edu.cn.)

learning, where datasets comprise sample-label pairs, and cannot be extended to MDP data (s, a, s', r, d) in continual reinforcement learning.

To adapt experience distillation to continual reinforcement learning and obtain experience with strong representation, we propose a robot continual reinforcement learning method based on experience consistency distillation, as shown in Fig. 1(b). Our method distills a large amount of Markov decision process data in continual reinforcement learning into a small set of distilled data. Fréchet Inception Distance (FID) [18] loss is utilized to constrain the distribution consistency between the experience before and after distillation. The policy of the robot is trained jointly on the distilled data and the new task data, and distills the old policy based on uncertainty metrics, better overcoming catastrophic forgetting. Compared to other replay-based methods like random sampling and storing coreset strategies, our method can effectively characterize the complete knowledge from the old task with distilled experience, reducing redundant information. Learning on just the distilled data can achieve comparable performance to the full old task data.

In summary, our contributions are as follows:

- We propose a continual reinforcement learning method utilizing experience consistency distillation. To our knowledge, it is the first approach to distill experience for robot continual reinforcement learning.
- We validate the efficiency of our method on the CW10 in Continual World [19]. Experimental results demonstrate that our approach outperforms other replay-based methods and reduces experience storage requirements.
- We conduct real-world manipulation experiments using a UR5e robot arm, showing the effectiveness and robustness of our method in physical real-world settings.

II. RELATED WORKS

A. Replay-based Continual Learning

Continual learning aims to enable models to continually acquire new skills over sequential tasks without retraining from scratch. Regularization-based methods impose constraints on model updates at the parameter [4] or gradient [20] level to combat catastrophic forgetting, but performance degrades with more tasks. Architecture methods expand neurons to accommodate the new task knowledge [9], but incur rapidly growing computational and data costs. Replay-based methods are the most widely used in continual learning, focusing on how to select buffer data and utilize it efficiently. For example, storing difficult samples for replay [5]. Or augmenting buffer samples during the new task learning [21]. However, current replay-based continual learning methods replay data with inadequate characterization. Therefore, we consider how to maximally retain knowledge about old tasks under a limited buffer budget.

B. Reinforcement Learning for Robotics

Reinforcement learning (RL), as an automatic control technology, has been widely applied in robotics [22]. In addition to robotic manipulation [23], RL has been widely used for navigation [24], unmanned system control [25] and multi-agent collaboration [26], etc. RL methods can be divided into on-policy RL and off-policy RL based on data sources. On-policy RL only learns from data sampled by the current policy [27]. Off-policy RL can learn from data sampled by any policies [28]. Since on-policy RL only uses data from the current policy, it has low sample efficiency and is unsuitable for robot learning. Therefore, we use Soft Actor-Critic (SAC) [29], an off-policy RL algorithm, as our baseline method.

C. Dataset Distillation

Dataset distillation aims to have models trained on distilled datasets match the performance of models trained on full datasets [17]. Current dataset distillation methods can be grouped into two categories. Target-domain validation-based distillation [17], [30], [31] update the distilled dataset using validation loss on the original dataset. Training-trajectory matching distillation match training trajectories on the distilled and original datasets to ensure similar performance. A representative target-domain validation-based method is KIP [32], which uses kernel ridge regression loss to update the distilled dataset. However, it ignores consistency of distributions between the distilled and real data, and is not suited for reinforcement learning. Our method employs Fréchet Inception Distance (FID) to ensure consistency of distributions before and after Markov decision process data distillation.

III. METHOD

Our method aims to efficiently history experience with strong representation for robot continual reinforcement learning. As shown in Fig. 2, in the experience distillation stage, we distill the state-action pairs in the buffer based on policy network and interact with the old environment to obtain the complete distilled MDP experience. In the policy learning stage, we train the policy by combining the distilled experience and current experience, and adopt uncertainty-based policy distillation to better exploit the distilled experience.

A. Problem Formulation

Continual Reinforcement Learning. Given a task sequence $1, 2, \ldots, T$, each task t has its own dataset $D_t (t \in 1, 2, \ldots, T)$. As data distributions differ across tasks, learning a new task inevitably induces catastrophic forgetting in a robot [1], rendering conventional deep learning techniques like fine-tuning ineffective. The overarching goal of continual reinforcement learning is to retain performance of the robot on old tasks $1, \ldots, t-1$ when learning a new task t.

Actor-Critic Method. We employ Soft Actor-Critic (SAC) [29] as our base RL algorithm. SAC is an off-policy actor-critic-based algorithm, where the actor represents the policy $\pi^t(a|s)$ that attempts to maximize the action value Q evaluated by the critic. The critic evaluates the value $Q(s, a)$ of taking an action a in a given state s, and aims to approximate the true action value function $Q^*(s, a)$.

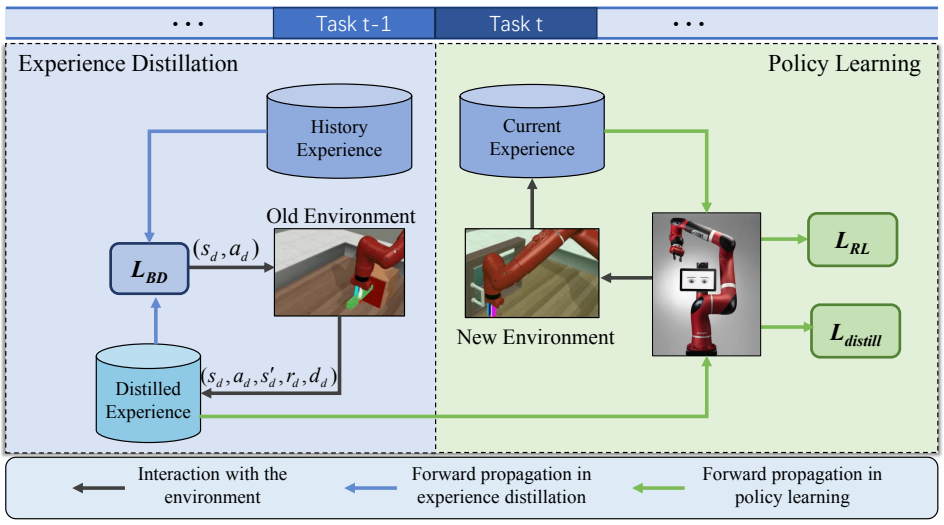


Fig. 2. An overview of our method. **Experience distillation(left):** We distill the experience of old tasks and interact with the environment to obtain distilled experience. **Policy learning(right):** We jointly train the policy network using the distilled experience and the new task data.

Algorithm 1: Experience Consistency Distillation for RL

Data: Original experience $(s_o, a_o, s'_o, r_o, d_o)$; Distillation buffer size B, which is much smaller than the original buffer size

Randomly initialize distilled data (s_d, a_d) of size B based on constraints of observation and action space.

foreach $i = 1, \ldots, N$ **do**

  Randomly initialize policy model $\pi^t(a|s)$.

  while *not converged* **do**

    Sample minibatch data $(\bar{s}_o, \bar{a}_o)$ from original experience.

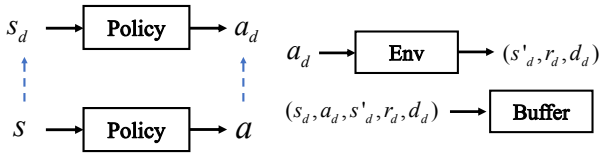
    Sample minibatch data $(\bar{s}_d, \bar{a}_d)$ from distilled data.

    Compute experience distillation loss based on Eq. 5.

    Back propagation to update the subset $(\bar{s}_d, \bar{a}_d)$ of (s_d, a_d).

Use distillation data $(\bar{s}_d, \bar{a}_d)$ to interact with the environment Env to obtain the next state s'_d, reward r_d, and done signal d_d.

Result: Distilled experience $(s_d, a_d, s'_d, r_d, d_d)$



(a) State-action distillation (b) Distilled experience

Fig. 3. The process of experience distillation for reinforcement learning.

B. Experience Distillation for Continual RL

In order to improve the data efficiency of history experience, we propose experience distillation for continual RL. As depicted in Fig. 3(a), s and a denote the state-action pairs before distillation, while s_d and a_d represent the distilled state-action pairs. To obtain the distilled experience, we first distill the state-action pairs based on the policy network. Our objective is to minimize the validation loss of the policy model trained on the distilled data compared to the original data. This indicates that the distilled data can effectively represent the original data, and the policy model trained on the distilled data can achieve comparable validation performance as the model trained on the original data. We iterate the above process until convergence. As depicted in Fig. 3(b), we next execute the distilled action a_d in the environment Env to obtain the next state s'_d, reward r_d, and done signal l_d. Finally, distilled experience tuples are stored in the buffer for experience replay during continual reinforcement learning.

As outlined in Algorithm 1, we distill the experience in the replay buffer based on kernel ridge regression. First, we initialize the distilled data (s_d, a_d), and randomly sample a batch of experience $(\bar{s}_o, \bar{a}_o)$ from the original buffer and a batch of data $(\bar{s}_d, \bar{a}_d)$ from the distilled experience. We then compute the kernel ridge regression loss according to:

$$\mathcal{L}(s_d, a_d) = \frac{1}{2}||a_o - K_{s_o s_d}(K_{s_d s_d} + \lambda I)^{-1} a_d||^2, \quad (1)$$

where K denotes an arbitrary kernel function. For any two samples u and v, $K_{uv} = K(u, v)$. We then update the subset $(\bar{s}_d, \bar{a}_d)$ of (s_d, a_d) by back propagating the loss $\mathcal{L}(s_d, a_d)$. This process is repeated until convergence to obtain the final distilled state-action pairs (s_d, a_d). The original kernel ridge regression formulation utilizes the neural tangent kernel (NTK) as the kernel function. However, the NTK $K(x_1, x_2) =< \frac{\mathrm{d}f_\theta(x_1)}{\mathrm{d}\theta}, \frac{\mathrm{d}f_\theta(x_2)}{\mathrm{d}\theta} >$ requires computing first-order derivatives, which is computationally expensive. To mitigate this issue, we instead adopt the conjugate kernel

(CK) defined as:

$$K(s_o, s_d) =< f(s_o, \theta), f(s_d, \theta) >= f(s_o, \theta) f(s_d, \theta)^\top, \tag{2}$$

$$K(s_d, s_d) =< f(s_d, \theta), f(s_d, \theta) >= f(s_d, \theta) f(s_d, \theta)^\top. \tag{3}$$

where θ represents the parameters of the policy network f_θ, and $f(\cdot, \theta)$ denotes the feature representations extracted from the input by the policy network f_θ.

C. Distribution Consistency Constraints

Although the above method can distill the MDP experience tuples (s, a, s', r, l) into a small dataset with strong representation, it does not explicitly consider distribution differences between the distilled and original experience. In reinforcement learning, the observation and action spaces often have bounded ranges, and the constraint may be lost after distillation. To address this problem, we propose FID loss as a regularization term to constrain the difference between the distilled and original experience distributions, inspired by techniques in image generation. Fréchet Inception Distance is a widely used evaluation metric in image generation that assesses the quality of synthesized images by measuring the feature-level divergence from real data. The FID loss is calculated as:

$$\begin{aligned} \mathcal{L}_{FID} &= d^2((\mu_d, \sigma_d), (\mu_r, \sigma_r)) \\ &= ||\mu_d - \mu_r||^2 + \text{Tr}(\Sigma_d + \Sigma_r - 2(\Sigma_d \Sigma_r)^{1/2}). \end{aligned} \tag{4}$$

where μ_r, Σ_r are the mean and covariance of feature representations extracted from the original real data, μ_d, Σ_d are those from the distilled data. Tr denotes the trace of the matrix. A smaller distance $d^2((\mu_d, \sigma_d), (\mu_r, \sigma_r))$ indicates greater similarity between the two distributions. By minimizing the FID loss, we regularize the distilled experience distribution to remain close to the original experience distribution. Therefore, the overall objective $\mathcal{L}_{ED}$ for experience distillation is:

$$\mathcal{L}_{ED} = \mathcal{L}_{KRR} + \lambda \cdot \mathcal{L}_{FID}. \tag{5}$$

Where L_{KRR} is the kernel ridge regression loss, and L_{FID} is the squared FID distance , weighted by a hyper-parameter λ. By minimizing this overall objective $\mathcal{L}$, we obtain distilled experience that effectively represents the original data while closely matching its underlying distribution.

D. Uncertainty-based Policy Distillation

The details of the proposed overall approach are presented in Algorithm 2. To better utilize the distilled experience and extract knowledge about old tasks, we train the new policy directly on the distilled data and also aim for its predictions on old task data to closely match those of the old policy. Formally, the policy model should satisfy:

$$\pi_\theta(\cdot|s_d^i) \approx \pi_{\theta_{old}}(\cdot|s_d^i), \forall s_d^i \in \mathcal{D}_{old}, \tag{6}$$

where π_θ denotes the new policy with parameters θ, $\pi_{\theta_{old}}$ denotes the old policy, and $\mathcal{D}_{old}$ represents the distilled old task experience.

Algorithm 2: Experience Distillation Continual SAC

Data: Update frequency K; Number of tasks T

Learn the first task using standard SAC algorithm.

foreach $t = 2, \ldots, T$ **do**

  Obtain the distilled old task experience using Algorithm 1.

  foreach $j = 1, 2, \ldots$ **do**

    Policy $\pi^t(a|s)$ interacts with environment Env to collect data into buffer.

    if $j\%K == 0$ **then**

      Update action-value function by SAC using the new task experience.

      Update policy $\pi^t(a|s)$ based on Eq. 10 using the new task experience and the distilled old task experience.

Result: Policy Network $\pi^t(a|s)$

In order to better estimate actions, our policy outputs a Gaussian distribution $\mathcal{N}(\mu^i, \sigma^i)$ over stochastic actions, where the final action is computed according to:

$$a^i = \mu^i + \sigma^i \odot z, \quad z \sim \mathcal{N}(0, I), \tag{7}$$

where $\odot$ denotes element-wise multiplication between two vectors. By outputting a distribution instead of a deterministic action, the policy can better capture uncertainty and exploit stochastically during training. The variance σ^i indicates the policy's confidence regarding each action dimension.

To perform policy distillation, we match the distributions by minimizing the uncertainty-based KL divergence between the old and the new policy:

$$\mathcal{L}_{distill} = \mathbb{E}_{s_d^i \sim \mathcal{D}_{old}}[KL(\mathcal{N}(\mu^i_{\theta_{old}}, \sigma^i_{\theta_{old}})||\mathcal{N}(\mu^i_\theta, \sigma^i_\theta))] \tag{8}$$

$$= \mathbb{E}_{s_d^i \sim \mathcal{D}_{old}}[-\frac{1}{2} + \log \frac{\sigma^i_\theta}{\sigma^i_{\theta_{old}}} + \frac{(\sigma^i_{\theta_{old}})^2 + (\mu^i_{\theta_{old}} - \mu^i_\theta)^2}{2(\sigma^i_\theta)^2}] \tag{9}$$

Intuitively, this divergence measures the discrepancy between the action distribution predicted by the new policy and the old policy on the distilled experience from the old task. By minimizing the KL divergence, we transfer the knowledge, including uncertainty information, from the old policy to the new policy. The new policy thereby learns to mimic the old policy's outputs on old task data.

Accordingly, the overall policy training objective is formalized as:

$$\mathcal{L}_{total} = \mathcal{L}_{RL} + \gamma \cdot \mathcal{L}_{distill}, \tag{10}$$

where $\mathcal{L}_{RL}$ denotes the standard reinforcement learning objective and γ represents a scalar hyperparameter that balances it with the distillation loss.

IV. EXPERIMENTS

In this section, we empirically analyze the advantages of our proposed method compared to other replay-based approaches on the Continual World and real scenarios.

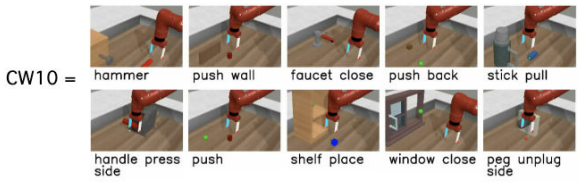


Fig. 4. The CW10 task sequence for continual reinforcement learning, containing 10 manipulation tasks.

TABLE I

QUANTITATIVE COMPARISON WITH OTHER REPLAY-BASED METHODS ON CW10.

Method	Average Success(↑)	P.t.Buffer Size(↓)	Forgetting(↓)
Finetune	0.10	0	0.74
Perfect Memory	0.29	1e6	0.03
iCaRL [34]	0.56	1e4	0.02
A-GEM [35]	0.13	1e4	0.73
GPM [36]	0.45	1e4	**-0.02**
OFSS [37]	0.69	1e4	0.01
MTT [38]	0.45	**1e3**	0.24
Ours	**0.78**	**1e3**	0.01
Full	0.84	1e6	0.00

A. Experimental Settings

Continual World. The simulation experiments are conducted in Continual World [19], which provides 10 continuous control robotic arm tasks from Meta-World [33], including hammer, push wall, faucet close, push back, etc. The observation space is 12-dimensional, comprised of the 3D Cartesian positions of the end effector, object 1, object 2, and the target. The action space is 4-dimensional, consisting of the 3D Cartesian position offset of the end effector and the gripper actuator delta. The reward is computed by a predefined finely-tuned reward function for each task. As illustrated in Fig. 4, we evaluate our method on the CW10 benchmark provided in Continual World, which is a sequential task curriculum of 10 tasks. The specifics of each task can be found in [33].

Baseline. We compare our method with the following other replay-based methods: (1) **Finetune** simply finetunes the model on the new task, representing the lower performance bound. (2) **iCaRL** [34] retains representative samples and finetunes the model. (3) **Perfect Memory** replays the full old task data. (4) **A-GEM** [35] updates the model while minimizing interference with old task knowledge. (5) **GPM** [36] makes model updates orthogonal to old task subspaces. (6) **OFSS** [37] updates the buffer with synthesized data using validation loss on real data, and replays it. (7) **Perfect Memory+Distillation** replays all old task data and performs policy distillation, representing the upper performance bound.

Metrics. We evaluate our method using average success, per-task buffer size and forgetting. Average success evaluates the performance across all tasks. Per-task buffer size measures the data efficiency in the replay buffer. Forgetting quantifies catastrophic forgetting.

Implementation Details. We use SAC as the base RL algorithm with 1 million steps per task. The distillation buffer size of per task is set to 1000. The weight of FID loss and policy distillation loss is 1.0 and 10^4, respectively. Three random seeds are utilized in our experiments. We follow the setting of CW10 for the metrics and multi-head models. After each task, the one-hot task identifier is updated for a new task. We simply use FIFO experience replay for buffer management. Adam optimizer [39] is used with a learning rate of 0.001.

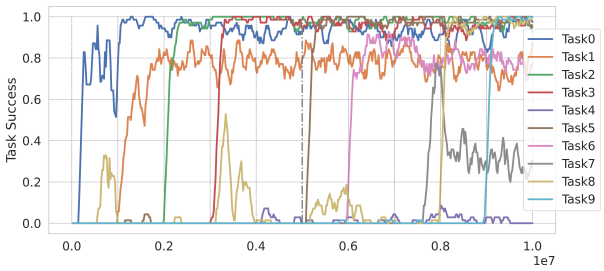


Fig. 5. Success of each task during continual learning.

B. Comparison with other methods

As presented in Table I, we compare our method with other replay-based methods on the CW10. *Full* refers to perfect memory combined with policy distillation, representing the upper bound performance for our method. Due to the setting of a small number of limited buffers in continual learning and privacy considerations, we cannot store the data of old tasks without restrictions. In contrast, our method only stores 1000 distilled samples per task in the buffer, greatly reducing the required buffer size. Compared to other methods, our method achieves better performance using only 1/1000 of the data in perfect memory, and approaches the upper bound. The per-task performance of our method is shown in Fig. 5. We also compare our method with the dataset distillation strategy based on matching training trajectories (MTT) [38] in Table I. The MTT strategy distills the dataset by matching training trajectories on the original data with those on the distilled data. The results demonstrate that our method significantly outperforms MTT, indicating that our method is better suited for interaction data scenarios without hard labels, as in reinforcement learning.

C. Detailed Analysis

Effectiveness of Experience Distillation. In the following experiments, we validate the effectiveness of experience distillation by comparing different buffer storage strategies, as presented in Table II. *Random* refers to randomly sampling 1000 samples per task into the buffer. With the same buffer size, the average success of our method is improved over the Random baseline by 22%. Moreover, our method only requires 1/1000 of the full data per task, while the performance only drops by 6% compared to the theoretical upper bound.

Ablation Study. We also verify the effectiveness of each component in our method, including uncertainty-based policy distillation, experience distillation, and FID loss, as shown in Table III. The results demonstrate that all components in our method improve the continual learning ability of

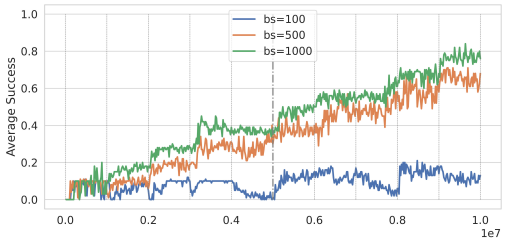


Fig. 6. Quantitative analysis for buffer size. The x-axis represents training steps on CW10.

TABLE II

THE IMPACT OF DIFFERENT BUFFER STORAGE STRATEGIES ON THE RESULTS.

Method	Average Success(↑)	P.t.Buffer Size(↓)	Forgetting(↓)
Random	0.56	**1e3**	0.02
Ours	**0.78**	**1e3**	**0.01**
Full	0.84	1e6	0.00

TABLE III

ABLATION STUDY ON EACH COMPONENT OF OUR METHOD.

Method	Average Success(↑)	P.t.Buffer Size(↓)	Forgetting(↓)
PD	0.56	**1e3**	0.02
PD+KRR	0.65	**1e3**	**0.01**
PD+KRR+FID	**0.78**	**1e3**	**0.01**

TABLE IV

AVERAGE SUCCESS OF FOUR TASKS IN SIMULATION AND REAL-WORLD.

Task	hammer	push-wall	push	shelf-place
Simulation	0.98	0.84	0.76	0.30
Real-world	0.90	0.70	0.60	0.30

the policy model. Thanks to the FID loss, the result of experience distillation is boosted by 13%, which indicates FID loss better constrains the data consistency before and after experience distillation, while preserving the constraints in the observation and action space of reinforcement learning.

Impact of Buffer Size. We also compare the effect of different per-task buffer sizes on policy continual learning, as depicted in Fig. 6. We experiment with buffer sizes of 100, 500, and 1000 samples per task. The results are consistent with the expectation that the larger buffer size leads to improved continual reinforcement learning performance. However, the increase in buffer size can result in exponentially increased computing and storage costs, and goes against the motivation of experience distillation. Given these tradeoffs, we uniformly use a per-task buffer size of 1000 samples in our experiments.

Impact of FID Weight. In Fig. 7, we further investigate the impact of the FID loss weight λ (fw in Fig. 7) on performance. We evaluate the performance when λ is 0.1, 1.0, 10, 100, 1000, respectively. The results indicate that 1.0 provides the optimal balance. With too small of an λ, the distilled data struggles to adequately match the original data distribution. Conversely, with a large λ, the strong regularization causes overfitting to the specific distributions before and after distillation, disregarding representations of the distilled data.

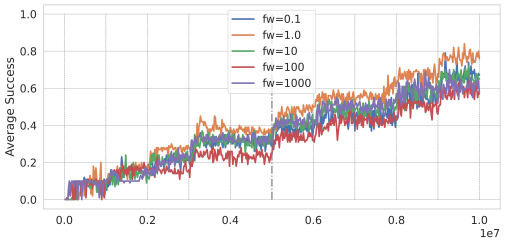


Fig. 7. Quantitative analysis on different FID loss weight.

Fig. 8. Real-world experiments. From top to bottom, each row corresponds to hammer, push, push-wall, and shelf-place respectively.

D. Real-world Experiments

Finally, we validate the efficiency of our method in real-world scenarios, as shown in Fig. 8. Due to the limitations of the experimental setup, some task settings are difficult to reproduce physically in the real world, e.g. closing window. Therefore, as shown in Table IV, we evaluate our method on four representative tasks, including pushing, push-wall, shelf-place and hammering. In the real-world experiments, we use the UR5e single-arm robot. We execute each task 10 times to get average success. ArUco markers are utilized to localize objects and targets, providing their 3D positions. These 3D coordinates together with the end-effector coordinates constitute the observation in real-world. The real-world results demonstrate that our method is effective and robust not only in simulation but also in real-world manipulation tasks.

V. CONCLUSION

In this work, we propose an experience consistency distillation continual reinforcement learning method for robotic manipulation tasks. Experience distillation can effectively strong-representative distilled data, greatly reducing required storage space. The proposed FID loss can effectively maintain the data distribution consistency before and after distillation. Uncertainty-based policy distillation can efficiently utilize the distilled experience. All our experimental results demonstrate the effectiveness of our method.

VI. ACKNOWLEDGEMENT

This work was supported in part by National Key R&D Program of China under grant No. 2021ZD0112700,NSFC under grant No.62125305, No. U23A20339, No. 62203348 and No.61973246.

REFERENCES

[1] T. Lesort, V. Lomonaco, A. Stoian, D. Maltoni, D. Filliat, and N. Díaz-Rodríguez, "Continual learning for robotics: Definition, framework, learning strategies, opportunities and challenges," *Information fusion*, vol. 58, pp. 52–68, 2020.

[2] B. Liu, X. Xiao, and P. Stone, "A lifelong learning approach to mobile robot navigation," *IEEE Robotics and Automation Letters*, vol. 6, no. 2, pp. 1090–1096, 2021.

[3] M. B. Hafez and S. Wermter, "Behavior self-organization supports task inference for continual robot learning," in *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2021, pp. 6739–6746.

[4] J. Kirkpatrick, R. Pascanu, N. Rabinowitz, J. Veness, G. Desjardins, A. A. Rusu, K. Milan, J. Quan, T. Ramalho, A. Grabska-Barwinska *et al.*, "Overcoming catastrophic forgetting in neural networks," *Proceedings of the national academy of sciences*, vol. 114, no. 13, pp. 3521–3526, 2017.

[5] X. Jin, J. Du, and X. Ren, "Gradient based memory editing for task-free continual learning," in *4th Lifelong Machine Learning Workshop at ICML 2020*, 2020.

[6] T. L. Hayes, G. P. Krishnan, M. Bazhenov, H. T. Siegelmann, T. J. Sejnowski, and C. Kanan, "Replay in deep learning: Current approaches and missing biological elements," *Neural computation*, vol. 33, no. 11, pp. 2908–2950, 2021.

[7] V. K. Verma, K. J. Liang, N. Mehta, P. Rai, and L. Carin, "Efficient feature transformations for discriminative and generative continual learning," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 13 865–13 875.

[8] A. Chaudhry, N. Khan, P. Dokania, and P. Torr, "Continual learning in low-rank orthogonal subspaces," *Advances in Neural Information Processing Systems*, vol. 33, pp. 9900–9911, 2020.

[9] A. A. Rusu, N. C. Rabinowitz, G. Desjardins, H. Soyer, J. Kirkpatrick, K. Kavukcuoglu, R. Pascanu, and R. Hadsell, "Progressive neural networks," *arXiv preprint arXiv:1606.04671*, 2016.

[10] J. Xu, J. Ma, X. Gao, and Z. Zhu, "Adaptive progressive continual learning," *IEEE transactions on pattern analysis and machine intelligence*, vol. 44, no. 10, pp. 6715–6728, 2021.

[11] H. Shin, J. K. Lee, J. Kim, and J. Kim, "Continual learning with deep generative replay," *Advances in neural information processing systems*, vol. 30, 2017.

[12] Z. Borsos, M. Mutny, and A. Krause, "Coresets via bilevel optimization for continual learning and streaming," *Advances in neural information processing systems*, vol. 33, pp. 14 879–14 890, 2020.

[13] R. Tiwari, K. Killamsetty, R. Iyer, and P. Shenoy, "Gcr: Gradient coreset based replay buffer selection for continual learning," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 99–108.

[14] J. Yoon, D. Madaan, E. Yang, and S. J. Hwang, "Online coreset selection for rehearsal-based continual learning," in *International Conference on Learning Representations*, 2021.

[15] R. Song, D. Liu, D. Z. Chen, A. Festag, C. Trinitis, M. Schulz, and A. Knoll, "Federated learning via decentralized dataset distillation in resource-constrained edge environments," in *2023 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2023, pp. 1–10.

[16] B. Zhao, K. R. Mopuri, and H. Bilen, "Dataset condensation with gradient matching," in *International Conference on Learning Representations*, 2020.

[17] T. Wang, J.-Y. Zhu, A. Torralba, and A. A. Efros, "Dataset distillation," *arXiv preprint arXiv:1811.10959*, 2018.

[18] M. Heusel, H. Ramsauer, T. Unterthiner, B. Nessler, and S. Hochreiter, "Gans trained by a two time-scale update rule converge to a local nash equilibrium," *Advances in neural information processing systems*, vol. 30, 2017.

[19] M. Wołczyk, M. Zajac, R. Pascanu, Ł. Kuciński, and P. Miłoś, "Continual world: A robotic benchmark for continual reinforcement learning," *Advances in Neural Information Processing Systems*, vol. 34, pp. 28 496–28 510, 2021.

[20] S. Lin, L. Yang, D. Fan, and J. Zhang, "Trgp: Trust region gradient projection for continual learning," in *The Tenth International Conference on Learning Representations*, 2022.

[21] J. Bang, H. Kim, Y. Yoo, J.-W. Ha, and J. Choi, "Rainbow memory: Continual learning with a memory of diverse samples," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, 2021, pp. 8218–8227.

[22] E. F. Morales, R. Murrieta-Cid, I. Becerra, and M. A. Esquivel-Basaldua, "A survey on deep learning and deep reinforcement learning in robotics with a tutorial on deep reinforcement learning," *Intelligent Service Robotics*, vol. 14, no. 5, pp. 773–805, 2021.

[23] H. Nguyen and H. La, "Review of deep reinforcement learning for robot manipulation," in *2019 Third IEEE International Conference on Robotic Computing (IRC)*. IEEE, 2019, pp. 590–595.

[24] K. Zhu and T. Zhang, "Deep reinforcement learning based mobile robot navigation: A review," *Tsinghua Science and Technology*, vol. 26, no. 5, pp. 674–691, 2021.

[25] H. Liu, B. Kiumarsi, Y. Kartal, A. Taha Koru, H. Modares, and F. L. Lewis, "Reinforcement learning applications in unmanned vehicle control: A comprehensive overview," *Unmanned Systems*, vol. 11, no. 01, pp. 17–26, 2023.

[26] J. K. Gupta, M. Egorov, and M. Kochenderfer, "Cooperative multi-agent control using deep reinforcement learning," in *Autonomous Agents and Multiagent Systems: AAMAS 2017 Workshops, Best Papers, São Paulo, Brazil, May 8-12, 2017, Revised Selected Papers 16*. Springer, 2017, pp. 66–83.

[27] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *arXiv preprint arXiv:1707.06347*, 2017.

[28] S. Fujimoto, H. Hoof, and D. Meger, "Addressing function approximation error in actor-critic methods," in *International conference on machine learning*. PMLR, 2018, pp. 1587–1596.

[29] 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.

[30] Y. Zhou, E. Nezhadarya, and J. Ba, "Dataset distillation using neural feature regression," *Advances in Neural Information Processing Systems*, vol. 35, pp. 9813–9827, 2022.

[31] I. Sucholutsky and M. Schonlau, "Soft-label dataset distillation and text dataset distillation," in *2021 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2021, pp. 1–8.

[32] T. Nguyen, R. Novak, L. Xiao, and J. Lee, "Dataset distillation with infinitely wide convolutional networks," *Advances in Neural Information Processing Systems*, vol. 34, pp. 5186–5198, 2021.

[33] T. Yu, D. Quillen, Z. He, R. Julian, K. Hausman, C. Finn, and S. Levine, "Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning," in *Conference on robot learning*. PMLR, 2020, pp. 1094–1100.

[34] S.-A. Rebuffi, A. Kolesnikov, G. Sperl, and C. H. Lampert, "icarl: Incremental classifier and representation learning," in *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, 2017, pp. 2001–2010.

[35] A. Chaudhry, M. Ranzato, M. Rohrbach, and M. Elhoseiny, "Efficient lifelong learning with a-gem," in *International Conference on Learning Representations*, 2018.

[36] G. Saha, I. Garg, and K. Roy, "Gradient projection memory for continual learning," in *International Conference on Learning Representations*, 2020.

[37] A. Rosasco, A. Carta, A. Cossu, V. Lomonaco, and D. Bacciu, "Distilled Replay: Overcoming Forgetting Through Synthetic Samples," in *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 13418 LNAI, 2022, pp. 104–117.

[38] G. Cazenavette, T. Wang, A. Torralba, A. A. Efros, and J.-Y. Zhu, "Dataset distillation by matching training trajectories," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 4750–4759.

[39] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.