Action Chunking Proximal Policy Optimization for Universal Robotic Dexterous Grasping

Sanghyun Hahn[1] and Jonghyun Choi[2]

***Abstract*— Universal dexterous grasping on different objects with diverse geometries is a crucial challenge in robotics for evolving towards human-like manipulation. In order to handle the high degrees of freedom (DoF) of dexterous hands, universal dexterous grasping methods adopt online Reinforcement Learning (RL) algorithms, particularly Proximal Policy Optimization (PPO), to learn action policies. Although PPO is a common choice, standard PPO itself often leads to insufficient exploration and slow policy improvement in complex, high-dimensional action spaces in dexterous hands. While recent RL algorithms exploit action chunking to enhance exploration, existing methods rely on high DoF Q-functions, making them unsuitable for universal dexterous grasping. To address these limitations, we propose *Action Chunking Proximal Policy Optimization* (ACPPO), a novel on-policy RL method whose policy outputs a chunk of actions from a single state. By reformulating the PPO objective over action chunks and using a standard state-value function as the critic, ACPPO retains PPO's simplicity while encouraging temporally coherent exploration and avoiding the curse of dimensionality. Our ACPPO outperforms PPO in grasp success rate by 5.7% and 4.6% on seen and unseen splits, respectively, with 9.4% faster training time.**

I. INTRODUCTION

Dexterous hands [1]–[3] are robotic grippers that aim to emulate the versatility of the human hand. While achieving human-level dexterity would unlock a whole new field of applications, dexterous grasping [4]–[6] still remains a significant challenge due to the high degrees of freedom (DoF) and diversity of the target objects.

Recent works [7]–[10] utilize Reinforcement Learning (RL) [11], especially Proximal Policy Optimization (PPO) [12] to learn universal dexterous grasping across diverse objects [13], varying in size and geometry. However, while PPO itself has proven to be excellent in robotic environments with small DoF [14]–[17], the algorithm alone struggles in the high-DoF dexterous grasping environment. Therefore, recent work on universal dexterous grasping augments PPO with additional techniques such as Curriculum Learning [7], Residual Learning and Mixture of Experts [9], Transformers [10], or Motion Objectives [8].

One promising direction for dexterous grasping is action chunking reinforcement learning (ACRL), where the policy outputs a short sequence of actions at each decision point. By committing multi-step commands, action chunking encourages temporally coherent exploration, reduces irregular movement near contact events, and allows efficient inference by using one policy forward pass for the next h environment steps.

Despite its potential, no prior ACRL method has been successfully applied to high-DoF dexterous grasping since existing designs are fundamentally incompatible with the scale of the problem. Prior action chunking methods learn a chunked action-value function $Q(s_t, a_{t:t+h-1})$ [18]–[20], where the action space dimension $h|\mathcal{A}|$ becomes impractical for dexterous hands with 20+ DoF. [21].

Based on this observation, we propose **Action Chunking Proximal Policy Optimization (ACPPO)**, the first method to apply action chunking in the domain of dexterous grasping. Our key insight is to integrate the action chunking mechanism directly into the actor and the on-policy surrogate objective, while completely avoiding intractable chunked Q-functions and offline data. Instead, ACPPO uses a simple, scalable state-value critic $V(s)$, allowing temporally coherent exploration while preserving the simplicity of PPO.

II. PRELIMINARIES

A. Markov Decision Processes

A standard reinforcement learning problem can be formally defined as a Markov Decision Process (MDP) [11]. The MDP is defined by the tuple $(\mathcal{S}, \mathcal{A}, p, r, \gamma)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, $p(s_{t+1}|s_t, a_t) : \mathcal{S} \times \mathcal{A} \to \mathcal{S}$ is the transition probability function, $r(s, a) : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function, and $\gamma \in [0, 1)$ is the discount factor. The behavior of the agent is determined by a stochastic policy $\pi(a|s)$.

In our universal dexterous manipulation setup, the states consists of the concatenation of robot states, PointNet features [22] of the target object, and the desired goal position of the object. The goal of the policy is to maximize the cumulative discounted reward, designed to guide the dexterous hand to grasp objects with diverse geometries.

B. Reinforcement Learning in Dexterous Grasping

Recently, reinforcement learning has been adopted in the domain of dexterous grasping since it can learn the high Degree-of-Freedom (DoF) action policies without requiring explicit access to the environmental dynamics. One major direction of exploiting reinforcement learning is focused on imitation learning [23]–[25], where policies are trained to mimic expert demonstrations. These methods focus on reducing distribution shift, and collecting demonstrations efficiently, for example through human videos.

Another line of work focuses on on-policy learning, where the policy is updated with its own rollouts. Our

[1] Department of Mechanical and Aerospace Engineering, Seoul National University. steve0221@snu.ac.kr

[2] Department of Electrical and Computer Engineering, Seoul National University. jonghyunchoi@snu.ac.kr

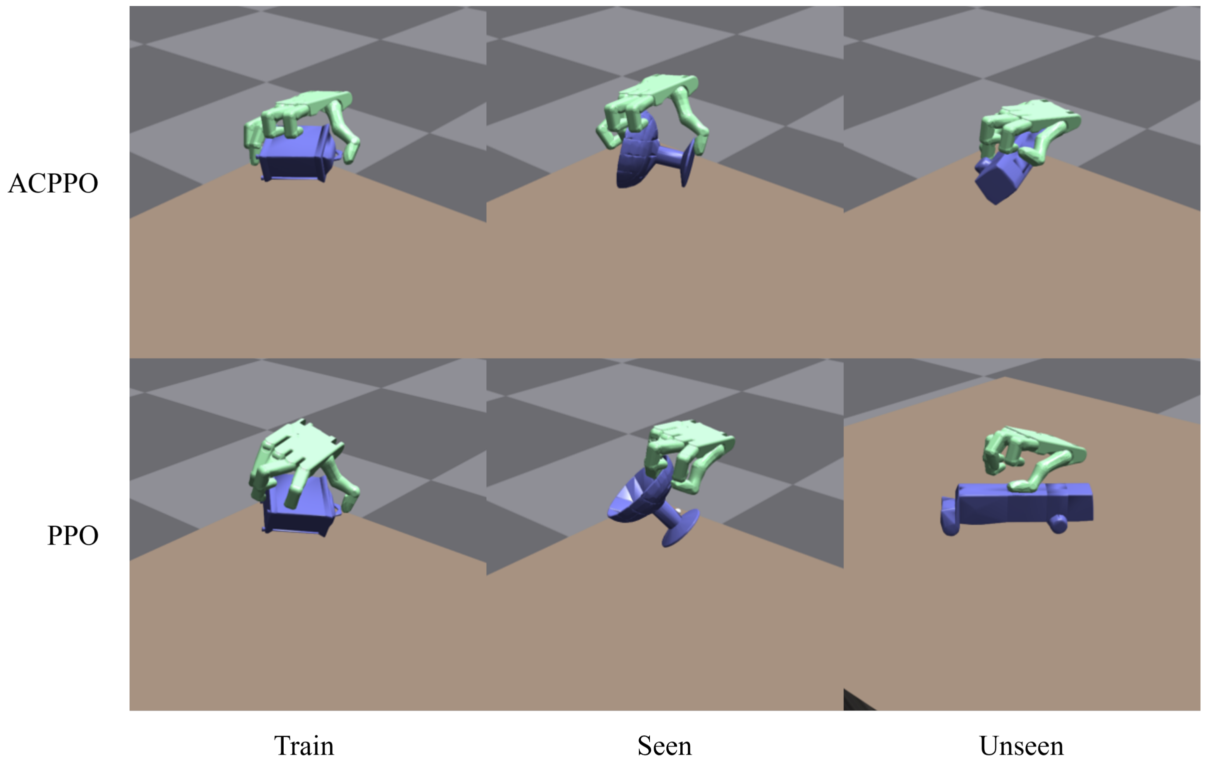

Fig. 1: Evaluation rollouts comparing ACPPO (top) and PPO (bottom). Seen and unseen test cases on seen-category and unseen-category objects. Trained on the same reward and environment, ACPPO produces more stable and successful grasps, outperforming standard PPO.

universal dexterous manipulation setup is mainly part of this group, since it is costly to collect demonstrations for thousands of different objects. PPO is widely used in this setting, since it offers stable training with clipped surrogate objectives. While PPO-based methods have shown promising results in universal dexterous grasping, they typically rely on additional mechanisms such as curriculum learning [7], residual learning [9], or transformer backbones [10]. This is primarily due to the high dimensional action space, which leads to the challenging exploration and the inefficiency of step-wise action learning in such complex systems.

C. Proximal Policy Optimization

Given a batch of trajectories collected by the behavior policy π_{θ_0} (typically the policy from the last update), PPO [12] performs multiple epochs of first-order updates on a clipped surrogate objective that controls the deviation of π_θ from π_{θ_0}.

Define the clipping minimum operator $\mathcal{C}_\epsilon$ and the per-step importance sampling ratio $\rho_t(\theta)$ as:

$$\mathcal{C}_\epsilon(\rho, A) := \min\left(\rho A, \text{clip}_{1-\epsilon}^{1+\epsilon}(\rho) A\right), \tag{1}$$

$$\rho_t(\theta) := \frac{\pi_\theta(a_t \mid s_t)}{\pi_{\theta_0}(a_t \mid s_t)}, \tag{2}$$

where $\text{clip}_{1-\epsilon}^{1+\epsilon}(\rho) := \max(1-\epsilon, \min(\rho, 1+\epsilon))$.

Let $\hat{A}_t$ denote an estimate of the advantage $A^{\pi_{\theta_0}}(s_t, a_t)$ (e.g., through GAE). PPO maximizes the clipped surrogate

$$\mathcal{L}_{\text{PPO}}(\theta) := \mathop{\mathbb{E}}_{\pi_{\theta_0}}\left[\mathcal{C}_\epsilon\big(\rho_t(\theta),\, \hat{A}_t\big)\right]. \tag{3}$$

This keeps ρ_t inside the trust region $[1-\epsilon, 1+\epsilon]$, following the behavior of TRPO [26] while retaining a simple first-order update. The full details of PPO are provided in V-A.

D. Action Chunking

Action chunking is a method that predicts and executes a sequence of actions rather than single-step commands [27]. In Imitation Learning, action chunking has been known to improve the robustness of learned policies, and handle non-Markovian behaviors of human demonstrations [28], [29]. Recent work have integrated action chunking into reinforcement learning, reporting that action chunking leads to efficient critic learning and encourages temporally coherent exploration [18]–[20].

However, these designs are impractical for dexterous grasping. Prior action chunking RL methods focus on learning the chunked action value function $Q(s_t, a_{t:t+h-1})$ with chunk length h. The action input dimension $\mathbb{R}^{h|\mathcal{A}|}$ is tractable for low-DoF robots such as manipulators, (DoF ≤ 7) but becomes unmanageable for dexterous hands with 20+ DoF [21]. In addition, several action chunking RL methods [19], [20] rely on an offline dataset, which is intractable for universal grasping. Universal grasping inevitably requires high dimensional visual features of the target objects, making it extremely costly to cover the state space as offline datasets. Architecturally, the transformer backbones used in some prior work [18] add expressibility to the chunked action value function, but also become computationally costly as the action dimension increases in dexterous hand settings.

To address these limitations, we propose Action Chunking Proximal Policy Optimization (ACPPO), a fully online action-chunked RL method that operates by learning the state value function $V(s)$. By avoiding the high dimensional chunked Q function, we mitigate the effect of $\mathbb{R}^{h|\mathcal{A}|}$ while retaining the benefits of action chunking.

III. METHOD

We implement our algorithm based on PPO, one of the most widely used on-policy reinforcement learning algorithms, which uses only the estimate of $V(s)$ as the critic. Our primary challenge is to reformulate the objective over action chunks while keeping the relative objective function unbiased, and maintaining a stable importance sampling ratio.

A. Naïve Action Chunking for PPO

A naïve way to add action chunking to PPO is to keep the step-wise actor unchanged and form a h-step open-loop likelihood ratio:

$$\rho^{\mathrm{n}}_{t,h}(\theta) := \prod_{k=0}^{h-1} \frac{\pi_\theta(a_{t+k}|s_{t+k})}{\pi_{\theta_0}(a_{t+k}|s_{t+k})}. \tag{4}$$

The corresponding objective function becomes:

$$\mathcal{L}^{(h)}_{\mathrm{n}}(\theta) := \mathop{\mathbb{E}}_{t}\Big[\mathcal{C}_\epsilon\Big(\rho^{\mathrm{n}}_{t,h}(\theta), A^{\pi_{\theta_0}}_{h}(s_t, a_{t:t+h-1})\Big)\Big]. \tag{5}$$

However, this objective function is biased. Since early actions within the chunk alter later actions through the states, multiplying step-wise ratios does not correct the state visitation inside the chunk. Furthermore, the product of ratios causes an exploding/vanishing behavior, making the objective almost always clipped.

B. Action Chunking PPO

For an accurate and more robust objective, we define a chunked actor $\pi(\cdot|s_{t:t+h-1})$ that predicts the next h sequence of actions from the current state. Also, we define the chunked advantage as follows:

$$\begin{aligned} &A^{\pi_{\theta_0}}(s_t, a_{t:t+h-1}) \\ &\quad= \sum_{k=0}^{h-1} \gamma^k r_{t+k} + \gamma^h V^{\pi_{\theta_0}}(s_{t+h}) - V^{\pi_{\theta_0}}(s_t). \end{aligned} \tag{6}$$

Since the actor outputs a h step sequence of actions, the importance sampling ratio now becomes:

$$\rho^{ch}_{t,h}(\theta) := \frac{\pi_\theta(a_{t:t+h-1}|s_t)}{\pi_{\theta_0}(a_{t:t+h-1}|s_t)}, \tag{7}$$

Algorithm 1 Action-Chunked PPO (ACPPO)

```
Init policy $\pi_\theta$, value $V_\phi$
while not converged do
    for each rollout do
        for $t = 0 : T$ do
            if $t \bmod h = 0$ then
                sample chunk $\mathbf{a}_{t:t+h-1} \sim \pi_\theta(\cdot|s_t)$
                cache $\log \pi_{\theta_0}, (\mu, \sigma)$
            Execute $a_t$, store $(s_t, a_t, r_t, d_t, V_\phi(s_t))$
    Compute GAE returns $\hat{R}_u$,
    Compute Chunked Advantage $A^{\pi_{\theta_0}}_h$
    for update epoch $e = 1{:}E$ do
        Sample $B$ batches from the chunk boundary
        Policy: maximize $\mathcal{L}^h_{\text{ACPPO}}(\theta)$
        Value: minimize $\mathbb{E}_B\left[\left(V_\phi(s_u) - \hat{R}_u\right)^2\right]$
        Update $(\theta, \phi)$ with LR $\alpha$
    Update behavior policy $\pi_{\theta_0} \leftarrow \pi_\theta$
return $\pi_\theta$
```

Using these formulas, we can exactly formulate the relative objective function for the policy gradient as:

$$\begin{aligned} &\mathcal{J}(\theta) - \mathcal{J}(\theta_0) \\ &= \mathop{\mathbb{E}}_{\substack{\tau\sim(p_0,\pi_\theta,p) \\ a'_t \sim \pi_{\theta_0}}}\left[\sum_{l=0}^{(T-1)/h} \gamma^{lh} \rho^{ch}_{lh,h}(\theta) A^{\pi_{\theta_0}}_{h}(s_{lh}, a_{lh:lh+h-1})\right] \end{aligned} \tag{8}$$

Eq. 8 is reduced into the relative objective function of standard PPO when $h = 1$. Replacing the chunked advantage with the chunked GAE estimate and applying importance ratio clipping, we obtain the objective function of ACPPO:

$$\mathcal{L}^h_{\text{ACPPO}}(\theta) := \mathop{\mathbb{E}}_{t}\Big[\mathcal{C}_\epsilon\big(\rho^{chunk}_{t,h}(\theta),\, A^{\pi_{\theta_0}}_{h}(s_t, a_{t:t+h-1})\big)\Big] \tag{9}$$

The ACPPO training loop is summarized in Algorithm 1, and the complete derivation of ACPPO is clarified in V-B.

C. Benefits of Action Chunking in Dexterous Manipulation

By training a policy to output action sequences, ACPPO provides several key advantages that are particularly impactful for high-DoF dexterous manipulation tasks. The core benefit, which has also been observed by previous action chunking studies [19], [27]–[29], is temporally coherent exploration.

Standard on-policy training often begins with high-frequency, jittery motions. Because these small random actions cancel each other out, this form of exploration effectively traps the agent within a small, local domain of the state space. Action chunking encourages the execution of smoother, multi-step maneuvers, allowing the policy to explore a much larger portion of the vast state space of the high DoF dexterous hand. While this broader exploration may result in a slower discovery of the global optimum direction, it allows the policy to gain a more comprehensive understanding of the environment's dynamics. This foundation leads the policy to build a better grasp in the long term, yielding higher final performance.

TABLE I: Experiment results of ACFQL, PPO, ACPPO on the DexGraspNet dataset.

Method	Train (%)	Test (%)		Training Time (s)
		Seen	Unseen	
ACFQL	79.1	77.2	79.0	**10,126**
PPO	82.3	82.1	82.0	11,702
ACPPO (Ours)	**87.9**	**87.8**	**86.6**	10,600

Beyond exploration, the chunked actor also benefits in inference. First, it improves inference speed by reducing the frequency of policy forward passes by h. Furthermore, the action chunks act like a low-pass filter on the commanded motion: when only part of the dexterous hand (e.g. one or two fingers) makes early contact with the objects, single-step policies often overreact with large whole-hand corrections. However, the chunked policy, on the other hand, continues the planned action sequence, avoiding oscillatory movement while the grasp is being completed.

IV. EXPERIMENT

A. Experiment Settings

We evaluate our method on the DexGraspNet [30] dataset, which consists of 5519 object instances from 133 object categories. Following UniDexGrasp++ [7], we use 3200 object instances for the training set, and construct two distinct test sets: (i) Seen-category objects, containing 141 unseen instances from categories included in the training set; (ii) Unseen-category objects, containing 100 objects from novel categories. For training ACPPO and the baseline PPO, we construct 3200 parallel simulation environments on IssacGym [31], and train for 20000 iterations on a single NVIDIA RTX A6000 GPU. For testing, we run 400 inferences for each object in the testing set. A rollout is considered successful if the object is placed at the desired location $z > 0$ after all action sequences have been executed. For fairness, we keep total environment steps and network sizes matched across methods and use $h = 2$ for all ACRL methods in the main results.

B. Experiment Results

We compare our method with the baseline reinforcement learning algorithms Action Chunking Flow-Q-Learning [19], Proximal Policy Optimization [12]. The chunk size of the action chunking methods were set to 2, which denotes the actor outputs two consecutive actions from a single state.

Table I shows that our method outperforms all baseline algorithms in universal dexterous grasping. Compared to PPO, ACPPO improves success by $+5.6\%$ on the train set and by $+5.7\%$/$+4.6\%$ on the seen/unseen test sets, respectively, while reducing training time by 9.4%.

Figure 2 illustrates the learning dynamics. While ACPPO rises more slowly in the early stage, it soon overtakes PPO and converges to a higher success rate. We attribute the initial lag to the temporally coherent exploration induced by action chunks, which spreads exploration around over

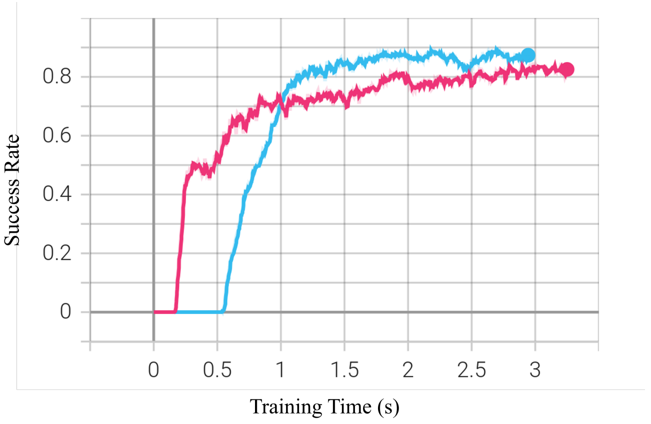

Fig. 2: Training curves of PPO (red) and ACPPO (blue); both trained for the same number of iterations.

a broader region of the state-space. As the critic improves, the actor receives more informative advantage estimates and accelerates learning. In terms of efficiency, ACPPO also benefits from fewer policy forward passes (one every h steps), which contributes to the observed speedup. Qualitatively, ACPPO rollouts exhibit fewer oscillations during pre-grasp alignment and stabler grasp poses as displayed in Fig. 1).

V. CONCLUSION

We introduced **Action Chunking Proximal Policy Optimization (ACPPO)**, an on-policy algorithm that predicts short action sequences while retaining a simple state-value critic. By reformulating the PPO surrogate over action chunks and training with a chunked advantage, ACPPO encourages temporally coherent exploration and reduces policy inference frequency. On DexGraspNet, ACPPO outperforms PPO and ACFQL across train, seen, and unseen splits, and reduces training time.

While our study focuses on state-based policies, distillation into vision-based policies via DAgger [32] is feasible and could extend our research to sim-to-real transfer. Furthermore, a key limitation of our work is that action chunking can reduce reactivity to sudden perturbations inside a chunk. While perturbations are not a major concern in simulated universal dexterous grasping environments, this can emerge as a major problem when adapting to real, dynamic settings. Future work could extend our approach to adaptive/learned chunk lengths, integrating curriculum or residual controllers, and real-to-sim transfer with tactile sensing to address this issue.

ACKNOWLEDGMENT

We would like to thank Taeyoung Kim for his valuable insight in Action Chunking. Furthermore, Large Language Models GPT 5 Thinking and Gemini 2.5 Pro helped polishing the writing and conducting literature reviews.

References

[1] J. L. Pons, R. Ceres, and F. Pfeiffer, “Multifingered dextrous robotics hand design and control: a review,” *Robotica*, vol. 17, no. 6, pp. 661–674, 1999.

[2] S. K. Sampath, N. Wang, H. Wu, and C. Yang, “Review on human-like robot manipulation using dexterous hands.” *Cogn. Comput. Syst.*, vol. 5, no. 1, pp. 14–29, 2023.

[3] A. M. Okamura, N. Smaby, and M. R. Cutkosky, “An overview of dexterous manipulation,” in *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1. IEEE, 2000, pp. 255–262.

[4] V. Kumar, A. Gupta, E. Todorov, and S. Levine, “Learning dexterous manipulation policies from experience and imitation,” *arXiv preprint arXiv:1611.05095*, 2016.

[5] M. Ciocarlie, C. Goldfeder, and P. Allen, “Dexterous grasping via eigengrasps: A low-dimensional approach to a high-complexity problem,” in *Robotics: Science and systems manipulation workshop-sensing and adapting to the real world*, 2007.

[6] M. Li, K. Hang, D. Kragic, and A. Billard, “Dexterous grasping under shape uncertainty,” *Robotics and Autonomous Systems*, vol. 75, pp. 352–364, 2016.

[7] W. Wan, H. Geng, Y. Liu, Z. Shan, Y. Yang, L. Yi, and H. Wang, “Unidexgrasp++: Improving dexterous grasping policy learning via geometry-aware curriculum and iterative generalist-specialist learning,” in *Proceedings of the IEEE/CVF International Conference on Computer Vision*, 2023, pp. 3891–3902.

[8] H. Zhang, S. Christen, Z. Fan, O. Hilliges, and J. Song, “Graspxl: Generating grasping motions for diverse objects at scale,” in *European Conference on Computer Vision*. Springer, 2024, pp. 386–403.

[9] Z. Huang, H. Yuan, Y. Fu, and Z. Lu, “Efficient residual learning with mixture-of-experts for universal dexterous grasping,” in *The Thirteenth International Conference on Learning Representations*, 2025.

[10] W. Wang, F. Wei, L. Zhou, X. Chen, L. Luo, X. Yi, Y. Zhang, Y. Liang, C. Xu, Y. Lu, *et al.*, “Unigrasptransformer: Simplified policy distillation for scalable dexterous robotic grasping,” in *Proceedings of the Computer Vision and Pattern Recognition Conference*, 2025, pp. 12 199–12 208.

[11] R. S. Sutton, A. G. Barto, *et al.*, *Reinforcement learning: An introduction*. MIT press Cambridge, 1998, vol. 1, no. 1.

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

[13] Y. Xu, W. Wan, J. Zhang, H. Liu, Z. Shan, H. Shen, R. Wang, H. Geng, Y. Weng, J. Chen, *et al.*, “Unidexgrasp: Universal robotic dexterous grasping via learning diverse proposal generation and goal-conditioned policy,” in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2023, pp. 4737–4746.

[14] A. Raffin, “Rl baselines3 zoo,” https://github.com/DLR-RM/rl-baselines3-zoo, 2020.

[15] A. Raffin, A. Hill, A. Gleave, A. Kanervisto, M. Ernestus, and N. Dormann, “Stable-baselines3: Reliable reinforcement learning implementations,” *Journal of Machine Learning Research*, vol. 22, no. 268, pp. 1–8, 2021. [Online]. Available: http://jmlr.org/papers/v22/20-1364.html

[16] M. Saeed, M. Nagdi, B. Rosman, and H. H. Ali, “Deep reinforcement learning for robotic hand manipulation,” in *2020 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)*. IEEE, 2021, pp. 1–5.

[17] D. Han, B. Mulyana, V. Stankovic, and S. Cheng, “A survey on deep reinforcement learning algorithms for robotic manipulation,” *Sensors*, vol. 23, no. 7, p. 3762, 2023.

[18] G. Li, D. Tian, H. Zhou, X. Jiang, R. Lioutikov, and G. Neumann, “Top-erl: Transformer-based off-policy episodic reinforcement learning,” in *7th Robot Learning Workshop: Towards Robots with Human-Level Abilities*.

[19] Q. Li, Z. Zhou, and S. Levine, “Reinforcement learning with action chunking,” in *The Exploration in AI Today Workshop at ICML 2025*.

[20] Y. Seo and P. Abbeel, “Reinforcement learning with action sequence for data-efficient robot learning,” in *CoRL 2024 Workshop on Whole-body Control and Bimanual Manipulation: Applications in Humanoids and Beyond*.

[21] R. Ding, Y. Qin, J. Zhu, C. Jia, S. Yang, R. Yang, X. Qi, and X. Wang, “Bunny-visionpro: Real-time bimanual dexterous teleoperation for imitation learning,” *arXiv preprint arXiv:2407.03162*, 2024.

[22] C. R. Qi, H. Su, K. Mo, and L. J. Guibas, “Pointnet: Deep learning on point sets for 3d classification and segmentation,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 652–660.

[23] Y. Qin, Y.-H. Wu, S. Liu, H. Jiang, R. Yang, Y. Fu, and X. Wang, “Dexmv: Imitation learning for dexterous manipulation from human videos,” in *European Conference on Computer Vision*. Springer, 2022, pp. 570–587.

[24] S. P. Arunachalam, S. Silwal, B. Evans, and L. Pinto, “Dexterous imitation made easy: A learning-based framework for efficient dexterous manipulation,” *arXiv preprint arXiv:2203.13251*, 2022.

[25] S. An, Z. Meng, C. Tang, Y. Zhou, T. Liu, F. Ding, S. Zhang, Y. Mu, R. Song, W. Zhang, *et al.*, “Dexterous manipulation through imitation learning: A survey,” *arXiv preprint arXiv:2504.03515*, 2025.

[26] J. Schulman, S. Levine, P. Abbeel, M. Jordan, and P. Moritz, “Trust region policy optimization,” in *International conference on machine learning*. PMLR, 2015, pp. 1889–1897.

[27] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, “Learning fine-grained bimanual manipulation with low-cost hardware,” in *ICML Workshop on New Frontiers in Learning, Control, and Dynamical Systems*.

[28] A. George and A. B. Farimani, “One act play: Single demonstration behavior cloning with action chunking transformers,” *arXiv preprint arXiv:2309.10175*, 2023.

[29] H. Bharadhwaj, J. Vakil, M. Sharma, A. Gupta, S. Tulsiani, and V. Kumar, “Roboagent: Generalization and efficiency in robot manipulation via semantic augmentations and action chunking,” in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 4788–4795.

[30] R. Wang, J. Zhang, J. Chen, Y. Xu, P. Li, T. Liu, and H. Wang, “Dexgraspnet: A large-scale robotic dexterous grasp dataset for general objects based on simulation,” *arXiv preprint arXiv:2210.02697*, 2022.

[31] V. Makoviychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, D. Hoeller, N. Rudin, A. Allshire, A. Handa, and G. State, “Isaac gym: High performance gpu-based physics simulation for robot learning,” 2021.

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

APPENDIX

A. Derivation of Proximal Policy Optimization

Define the importance ratio $\rho_t(\theta)$ and advantage as $A^\pi(s_t, a_t)$:

$$\rho_t(\theta) := \frac{\pi_\theta(a_t|s_t)}{\pi_{\theta_0}(a_t|s_t)}, \tag{10}$$

$$A^\pi(s_t, a_t) := Q^\pi(s_t, a_t) - V^\pi(s_t). \tag{11}$$

The relative objective function $\mathcal{J}(\theta) - \mathcal{J}(\theta_0)$ for the policy gradient can be formulated as follows:

$$\begin{aligned}
&\mathcal{J}(\theta) - \mathcal{J}(\theta_0) \\
&= \mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_\theta,p)}\left[\sum_{t=0}^{T-1} \gamma^t r_t + \gamma V^{\pi_{\theta_0}}(s_{t+1}) - V^{\pi_{\theta_0}}(s_t)\right] \\
&= \mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_\theta,p)}\left[\sum_{t=0}^{T-1} \gamma^t (Q^{\pi_{\theta_0}}(s_t, a_t) - V^{\pi_{\theta_0}}(s_t))\right] \qquad (12) \\
&= \mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_\theta,p) a'_t\sim\pi_{\theta_0}}\left[\sum_{t=0}^{T-1} \gamma^t \frac{\pi_\theta(a'_t|s_t)}{\pi_{\theta_0}(a'_t|s_t)} A^{\pi_{\theta_0}}(s_t, a'_t)\right]. \qquad (13)
\end{aligned}$$

Since the expectation that depends on the current policy θ is hard to utilize, define the surrogate objective function:

$$\mathcal{K}(\theta; \theta_0) = \mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_{\theta_0},p)}\left[\sum_{t=0}^{T-1} \gamma^t \rho_t(\theta) A^{\pi_{\theta_0}}(s_t, a'_t)\right]. \tag{14}$$

This surrogate objective function is accurate up to the first order, i.e.:

$$\nabla_\theta \mathcal{J}(\theta)\Big|_{\theta=\theta_0} = \nabla_\theta \mathcal{K}(\theta; \theta_0)\Big|_{\theta=\theta_0}. \tag{15}$$

PPO maximizes the surrogate objective function while keeping the current policy θ close to the driving policy θ_0 by clipping the importance ratio. The corresponding objective function can be written as:

$$\mathcal{L}_{\text{clip}}(\theta) := \mathop{\mathbb{E}}_{\pi_{\theta_0}}\left[\mathcal{C}_\epsilon\Big(\rho_t(\theta), A_t^{\pi_{\theta_0}}\Big)\right]. \tag{16}$$

While using $A^{\pi_{\theta_0}}$ directly in the PPO term has no bias, it causes variance as a tradeoff. Therefore, we use a Generalized Advantage Estimate (GAE) $\hat{A}_t^{\text{GAE}(\lambda)}$ with temporal-difference residuals δ_t:

$$\delta_t^{(\phi)} := r_t + \gamma V_\phi(s_{t+1}) - V_\phi(s_t) \tag{17}$$

$$\hat{A}_t^{\text{GAE}(\lambda)} := \sum_{l=0}^{\infty} (\gamma\lambda)^l \, \delta_{t+l}. \tag{18}$$

The GAE estimators are now biased, but have smaller variance compared to $A^{\pi_{\theta_0}}$. Using GAE, we finalize the PPO objective function as:

$$\mathcal{L}_{\text{PPO}}(\theta) := \mathop{\mathbb{E}}_{\pi_{\theta_0}}\left[\mathcal{C}_\epsilon\Big(\rho_t(\theta), \hat{A}_t^{\text{GAE}(\lambda)}\Big)\right]. \tag{19}$$

B. Action Chunking PPO

Define the chunked importance sampling ratio as:

$$\rho_{t,h}^{ch}(\theta) := \frac{\pi_\theta(a_{t:t+h-1}|s_t)}{\pi_{\theta_0}(a_{t:t+h-1}|s_t)}. \tag{20}$$

Starting from 12, we apply a h-step tower property:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_\theta,p)}\left[\sum_{t=0}^{T-1} \gamma^t (Q^{\pi_{\theta_0}}(s_t, a_t) - V^{\pi_{\theta_0}}(s_t))\right] \\
&= \mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_\theta,p)}\left[\sum_{t=0}^{T-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a_t)\right] \\
&= \mathop{\mathbb{E}}_{\tau^{(h)}}\left[\sum_{t=0}^{h-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a_t) + \mathop{\mathbb{E}}_{\pi_\theta}\left[\sum_{t=h}^{T-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a_t)\middle|\tau^{(h)}\right]\right] \\
&= \mathop{\mathbb{E}}_{\substack{s_0\sim p_0, s_{t+1}\sim p(\cdot|s_t,a_t)\\ a_{0:h-1}\sim\pi_\theta(\cdot|s_0)\\ a'_{0:h-1}\sim\pi_{\theta_0}(\cdot|s_0)}}\left[\frac{\pi_\theta(a'_{0:h-1}|s_0)}{\pi_{\theta_0}(a'_{0:h-1}|s_0)} \sum_{t=0}^{h-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a'_t)\right. \\
&\qquad\qquad \left. + \mathop{\mathbb{E}}_{\pi_\theta}\left[\sum_{t=h}^{T-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a_t)\middle|\tau^{(h)}\right]\right] \\
&= \mathop{\mathbb{E}}_{\substack{\tau\sim(p_0,\pi_\theta,p)\\ a'_t\sim\pi_{\theta_0}}}\left[\sum_{l=0}^{(T-1)/h} \left[\rho_{lh,h}^{ch}(\theta) \sum_{t=lh}^{(l+1)h-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a'_t)\right]\right]
\end{aligned}$$

This formula is exact, regarding that all actions inside the chunked are conditioned on the same state, independent of each other. However, simply using the cumulative discounted advantage $\sum \gamma^t A(s, a)$ causes a high variance. Instead, we define a chunked advantage:

$$\begin{aligned}
&A^{\pi_{\theta_0}}(s_t, a_{t:t+h-1}) \\
&\qquad = \sum_{k=0}^{h-1} \gamma^k r_{t+k} + \gamma^h V^{\pi_{\theta_0}}(s_{t+h}) - V^{\pi_{\theta_0}}(s_t).
\end{aligned} \tag{21}$$

Note that

$$\mathbb{E}_p\Big[\sum_{k=0}^{h-1} \gamma^k A^{\pi_{\theta_0}}(s_{t+k}, a_{t+k})\Big] = \mathbb{E}_p\Big[A_h^{\pi_{\theta_0}}(s_t, a_{t:t+h-1})\Big].$$

Using the chunked advantage, we can rewrite:

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\substack{\tau\sim(p_0,\pi_\theta,p)\\ a'_t\sim\pi_{\theta_0}}}\left[\sum_{l=0}^{(T-1)/h} \left[\rho_{lh,h}^{ch}(\theta) \sum_{t=lh}^{(l+1)h-1} \gamma^t A^{\pi_{\theta_0}}(s_t, a'_t)\right]\right] \\
&= \mathop{\mathbb{E}}_{\substack{\tau\sim(p_0,\pi_\theta,p)\\ a'_t\sim\pi_{\theta_0}}}\left[\sum_{l=0}^{(T-1)/h} \gamma^{lh} \rho_{lh,h}^{ch}(\theta) A_h^{\pi_{\theta_0}}(s_{lh}, a_{lh:lh+h-1})\right] \\
&\approx \mathop{\mathbb{E}}_{\tau\sim(p_0,\pi_{\theta_0},p)}\left[\sum_{l=0}^{(T-1)/h} \gamma^{lh} \rho_{lh,h}^{ch}(\theta) A_h^{\pi_{\theta_0}}(s_{lh}, a_{lh:lh+h-1})\right]
\end{aligned}$$

The final $\approx$ is exactly the approximation we make for the surrogate objective function in 14. For further reducing the variance, we can replace the chunked advantage with GAE estimates:

$$\widehat{A}_{t:h}^{(\lambda)} := \sum_{k=0}^{T-1} (\gamma\lambda)^k \, \delta_{t+k}^{\pi_{\theta_0}}. \tag{22}$$