Morphological Symmetry-aware Generalized Policy Network for Deep Reinforcement Learning

Ryo Hakoda [1,*], **Yubin Liu** [1], **Matthew Hwang** [1], **Yoshihiro Sato** [2], **Jun Takamatsu** [3], **Katsushi Ikeuchi** [3] **and Takeshi Oishi** [1]

[1] *Institute of Industrial Science, The University of Tokyo, Tokyo, Japan*
[2] *Engineering Faculty, Kyoto University of Advanced Science, Kyoto, Japan*
[3] *Microsoft, Redmond, USA*

Correspondence*:
Ryo Hakoda
r-hakoda@cvl.iis.u-tokyo.ac.jp

ABSTRACT

Exploiting the morphological symmetry of robotic systems, such as humanoid and quadruped robots, is a promising direction for improving robot learning. In deep reinforcement learning (DRL) for robot control, prior studies have leveraged such symmetry to improve learning efficiency through data augmentation, equivariant multilayer perceptrons (EMLPs), and multi-agent reinforcement learning (MARL) formulations. However, DRL training is inherently unstable, as the data distribution strongly depends on exploration, which is driven by stochasticity in the environment. To address this issue, we propose a symmetry-assisted, general-purpose DRL framework for morphologically symmetric robots that enables stable and robust learning. The framework models the environment as a symmetric Markov decision process (MDP) and constructs a full-body policy from a single-sided base policy using symmetry operators. We further propose a symmetric PPO objective with a coupled importance-sampling ratio. This objective aligns the policy optimization process with the imposed symmetry and serves as a principled alternative to MAPPO-style multi-agent formulations. Experimental results demonstrate that the proposed method outperforms existing approaches on most symmetric tasks, while still maintaining performance comparable to or better than standard PPO on asymmetric tasks, where symmetry is less directly exploitable.

Keywords: deep reinforcement learning, morphological symmetry, manipulation, legged locomotion, humanoid robots, quadruped robots

1 INTRODUCTION

Deep reinforcement learning (DRL) has recently demonstrated remarkable capabilities in controlling robots with high-degree-of-freedom (DoF), enabling behaviors that range from agile locomotion (Zhang et al., 2024) to dexterous manipulation (Luo et al., 2025). These advances are primarily fueled by the representational capacity of deep neural networks and scalable on-policy optimization techniques (such as PPO), which enable policies to model highly nonlinear dynamics. Despite this, the large dimensionality of the state and action spaces, combined with discrete contact events and underactuated dynamics, makes the learning of such systems fundamentally challenging (Lu et al., 2023; Barakat et al., 2023). As a result, training policies for these robots remains inefficient and prone to instability, especially for robots with many joints that operate under complex contact interactions.

Although Proximal Policy Optimization (PPO) (Schulman et al., 2017) is one of the most widely used learning algorithms in robotics, it can exhibit instability when applied to the training of humanoid and quadruped robots. PPO's clipped objective provides a simple and effective mechanism for stabilizing on-policy updates, making it a standard baseline in many robot-learning studies. However, high actuation redundancy amplifies gradient variance, allowing exploration-induced asymmetries to accumulate and often resulting in biased behaviors.

Symmetry has long played a central role in analytical mechanics and motion generation (Ghaffari et al., 2022; Dong et al., 2023; Ordonez-Apraez et al., 2023; Apraez et al., 2025). Many robotic systems, including humanoid and quadruped robots, possess morphological symmetries, with left–right reflection being particularly notable. In recent years, these morphological symmetries have been incorporated into DRL by formulating the environment as a symmetric Markov decision process (MDP) (Zinkevich and Balch, 2001). In this context, a natural direction for exploiting symmetry is to embed it directly into the learning problem rather than treating it as a post-hoc regularization or data-level heuristic.

Many approaches have been proposed to improve performance and training efficiency by exploiting the symmetry of robots; yet, each comes with its own limitations and potential drawbacks. Loss-based methods introduce auxiliary penalties to discourage asymmetric outputs (Yu et al., 2018; Kasaei et al., 2021; Nguyen et al., 2024; Yu et al., 2024). Such soft constraints are sensitive to the choice of penalty weights and can destabilize policy updates. Data-augmentation methods improve sample efficiency by adding symmetric counterparts of observed state–action pairs (Lin et al., 2020; Mittal et al., 2024; Nguyen et al., 2024; Wang et al., 2025; Bao et al., 2025). However, these approaches do not guarantee that the learned policy itself satisfies the desired symmetry. Architectural approaches enforce symmetry at the network architecture level, either by explicitly mirroring the robot's configuration and formulating the problem as a multi-agent RL problem (Abdolhosseini et al., 2019; Sonmez et al., 2024; Yan et al., 2024; Li et al., 2025), or by using symmetry-aware network architectures such as group-equivariant multilayer perceptrons (EMLPs) (Finzi et al., 2021; Kirsch et al., 2022; Wang et al., 2022; Liu et al., 2023; Su et al., 2024; Huang et al., 2023). Yet, these approaches do not guarantee stable policy updates against asymmetric samples that inevitably occur in random exploration in the environment.

In this paper, we present a symmetry-based DRL framework that leverages morphological symmetry at both the level of policy construction and optimization. We formulate the environment as a symmetric MDP equipped with left–right symmetry operators defined over the state and action spaces. Based on this formulation, we decompose the robot's configuration and action spaces into left and right components and train a single base policy that produces actions for one side of the robot, in line with MARL-based formulations (Abdolhosseini et al., 2019; Yan et al., 2024). Full-body actions are obtained by applying the symmetry operators to the base policy actions produced from mirrored observations. This approach ensures that the resulting policy is equivariant with respect to the prescribed symmetry, while reducing the effective action dimensionality and maintaining a network architecture as simple as a standard MLP.

Beyond the policy architecture, we argue that the learning objective itself must respect the joint symmetry of the robot. To this end, we derive an objective that treats the left and right actions as a single joint policy update, rather than as two loosely coupled agents, as in standard MAPPO-style MARL(Yu et al., 2022). Concretely, we define a coupled importance-sampling ratio based on the full symmetric policy and apply the PPO clipping to this ratio, ensuring that both sides of the robot are updated in a coordinated, symmetry-consistent manner. This stands in contrast to MARL formulations that share parameters across sides but perform per-side clipping, which can lead to partially inconsistent updates when the collected experience is asymmetric.

Table 1. Summary of the main spaces, variables, and objectives.

Symbol	Meaning
$\mathcal{S}, \mathcal{A}, \mathcal{Q}$	State, action, and joint-configuration spaces.
$\mathcal{Q}_{\mathrm{l}}, \mathcal{Q}_{\mathrm{r}}$	Left and right joint-configuration spaces.
$\mathcal{A}_{\mathrm{l}}, \mathcal{A}_{\mathrm{r}}$	Left and right action spaces.
$\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{q}$	State, action, and joint configuration.
$\boldsymbol{q}_{\mathrm{l}}, \boldsymbol{q}_{\mathrm{r}}$	Left and right components of the joint configuration.
$\boldsymbol{a}_{\mathrm{l}}, \boldsymbol{a}_{\mathrm{r}}$	Left and right action components.
π_θ	Full-body stochastic policy.
$\hat{\pi}_\theta$	Base policy defined on the left action space.
V_ϕ, V_{sym}	Critic network and its symmetrized output.
ρ_θ	Standard PPO importance-sampling ratio.
$\rho_{\mathrm{l}}, \rho_{\mathrm{r}}$	Per-side importance-sampling ratios.
ρ_{sym}	Coupled symmetric importance-sampling ratio.
$A, A_{\mathrm{l}}, A_{\mathrm{r}}$	Full and per-side advantage functions.
$J_{\mathrm{PPO}}, J_{\mathrm{MARL}}, J_{\mathrm{sym}}$	Standard PPO, MARL-style PPO, and proposed symmetric PPO objectives.

Table 2. Summary of symmetry-related operators and maps.

Symbol	Domain / Codomain	Meaning
$\mathcal{M}_s$	$\mathcal{S} \to \mathcal{S}$	Symmetry operator on the state/observation space.
$\mathcal{M}_a$	$\mathcal{A} \to \mathcal{A}$	Symmetry operator on the full action space.
$\mathcal{M}_q$	$\mathcal{Q} \to \mathcal{Q}$	Symmetry operator on the full joint configuration.
$\mathcal{M}_{\mathrm{split}}$	bijection	Split operator on either configurations or actions: $\mathcal{X} \to \mathcal{X}_{\mathrm{l}} \times \mathcal{X}_{\mathrm{r}}$, where $\mathcal{X} \in \{\mathcal{Q}, \mathcal{A}\}$.
$\hat{\mathcal{M}}_q$	$\mathcal{Q}_{\mathrm{l}} \to \mathcal{Q}_{\mathrm{r}}$	Left-to-right configuration correspondence map.
$\hat{\mathcal{M}}_a$	$\mathcal{A}_{\mathrm{l}} \to \mathcal{A}_{\mathrm{r}}$	Left-to-right action correspondence map.
$P_{\mathrm{intr}}, P_{\mathrm{extr}}$	permutation operators	Permute intrinsic and extrinsic observation entries.
$F_{\mathrm{intr}}, F_{\mathrm{extr}}$	value-flipping operators	Flip signs to intrinsic and extrinsic observation entries.

Our main contributions are summarized as follows:

- We formulate a symmetry-based general-purpose DRL framework for morphologically symmetric robots. Our approach models the environment as a symmetric MDP and explicitly constructs a full-body policy from a single-sided base policy via symmetry operators.
- We derive a symmetric PPO objective based on a coupled importance-sampling ratio for the full policy. This formulation aligns the optimization procedure with the imposed symmetry and can be regarded as a principled alternative to MAPPO-style multi-agent objectives for symmetric robots.
- We conduct extensive empirical evaluation on ten symmetric and six asymmetric tasks across five robotic platforms, including real humanoid and quadruped robots, comparing PPO (Schulman et al., 2017), data augmentation (Mittal et al., 2024), EMLP (Su et al., 2024), MARL (Yan et al., 2024), and our method.

2 METHODOLOGY

In this section, we first present brief preliminaries on DRL, the morphological symmetry of robotic systems, and the corresponding symmetric MDP formulation, and then propose a symmetry-based DRL framework

that exploits these structures in both policy construction and optimization. For clarity, Tables 1 and 2 summarize the main symbols, spaces, and symmetry-related operators used throughout this section.

2.1 Preliminary

2.1.1 Problem Formulation and PPO Objective

In deep reinforcement learning (DRL), an agent learns optimal actions from experience obtained through interactions with the environment. The environment is modeled as a Markov decision process (MDP) (Sutton and Barto, 1998), represented as a tuple $(\mathcal{S}, \mathcal{A}, r, T, p_0)$, where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the reward function. The transition density function $T : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \to [0, 1]$ specifies the transition probability, where $T(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a})$ denotes the probability of transitioning to state $\boldsymbol{s}'$ from state $\boldsymbol{s}$ when taking action $\boldsymbol{a}$. The initial state distribution $p_0 : \mathcal{S} \to [0, 1]$ gives the probability that the Markov process starts in state s.

The agent interacts with the environment by selecting actions according to a stochastic policy $\pi : \mathcal{S} \times \mathcal{A} \to [0, 1]$, where $\pi(\boldsymbol{a} \mid \boldsymbol{s})$ denotes the probability of taking action $\boldsymbol{a}$ in state $\boldsymbol{s}$. In robot control tasks, the action space is typically continuous and can be written as $\mathcal{A} \subset \mathbb{R}^n$, where n is the dimension of the action space. In this setting, we model the policy π_θ as a Gaussian

$$\pi_\theta(\boldsymbol{a} \mid \boldsymbol{s}) = \mathcal{N}\big(\boldsymbol{a}\,;\, \mu_\theta(\boldsymbol{s}), \sigma_\theta(\boldsymbol{s})\big),$$

where $\mu_\theta(\boldsymbol{s}) \in \mathbb{R}^n$ and $\sigma_\theta(\boldsymbol{s}) \in \mathbb{R}^n_{>0}$ are the mean and standard-deviation vectors, parameterized by θ.

In practice, we optimize the policy parameters θ using Proximal Policy Optimization (PPO) (Schulman et al., 2017). In PPO, the policy network is referred to as the *actor* network, and it is updated by maximizing a clipped surrogate objective based on an importance-sampling ratio $\rho_\theta(\boldsymbol{s}, \boldsymbol{a})$.

$$\rho_\theta(\boldsymbol{s}, \boldsymbol{a}) \coloneqq \frac{\pi_\theta(\boldsymbol{a} \mid \boldsymbol{s})}{\pi_{\text{old}}(\boldsymbol{a} \mid \boldsymbol{s})},$$

where π_{old} denotes the actor policy used to collect data. The standard PPO objective is defined with the advantage function $A(\boldsymbol{s}, \boldsymbol{a})$, as follows:

$$J_{\text{PPO}}(\theta) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a})\sim\mathcal{D}}\Big[\min\big(\rho_\theta(\boldsymbol{s}, \boldsymbol{a})A(\boldsymbol{s}, \boldsymbol{a}), \text{clip}(\rho_\theta(\boldsymbol{s}, \boldsymbol{a}), 1 - \epsilon, 1 + \epsilon)A(\boldsymbol{s}, \boldsymbol{a})\big)\Big],$$

where $\mathcal{D}$ is the dataset collected under π_{old}, and $\epsilon > 0$ is a hyperparameter for the clipping function $\text{clip}()$ that controls the update range. In practice, this surrogate is combined with a value-function loss and an entropy bonus to stabilize learning.

2.1.2 Robot Morphology-based Symmetry Groups

Many robotic systems, such as humanoid and quadruped robots, can be modeled using a left–right reflection symmetry group, denoted by $\mathbb{G} \coloneqq \mathbb{C}_2 = \langle g \mid g^2 = e \rangle$, where e denotes the identity element and g is the generator representing the left–right reflection (Finzi et al., 2021). This means that their links and joints are arranged symmetrically with respect to the sagittal plane of the body. The generator g induces symmetry operators on the state and action spaces, which we denote by $\mathcal{M}_s : \mathcal{S} \to \mathcal{S}$ and $\mathcal{M}_a : \mathcal{A} \to \mathcal{A}$, respectively. Since $\mathbb{C}_2$ is a two-element group with $g^2 = e$, these operators are involutions, i.e., $\mathcal{M}_s^2 = \text{Id}$ and $\mathcal{M}_a^2 = \text{Id}$, and hence $\mathcal{M}_s^{-1} = \mathcal{M}_s$ and $\mathcal{M}_a^{-1} = \mathcal{M}_a$. In practice, $\mathcal{M}_s$ and $\mathcal{M}_a$ can be implemented as

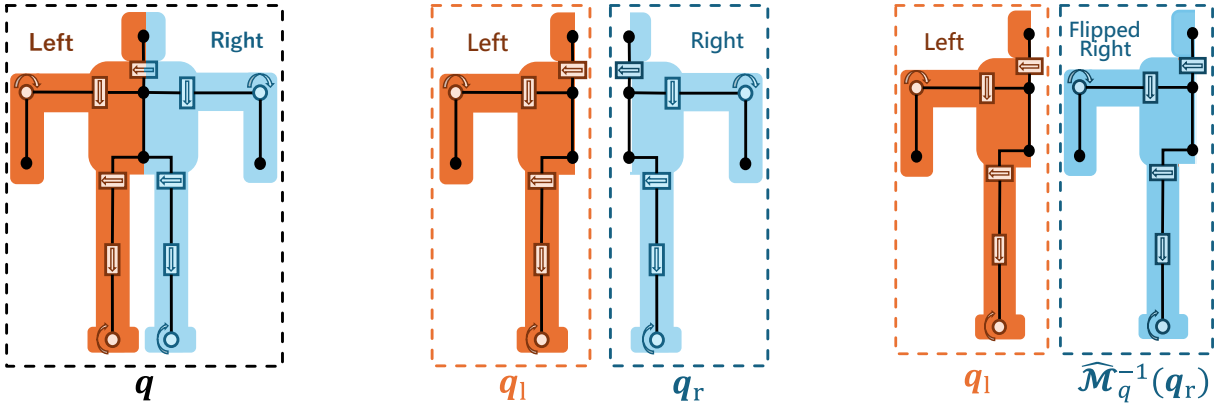

Figure 1. Example of configuration-space decomposition. **Left**: a full-body humanoid with morphological symmetry, whose joint state is denoted by $\boldsymbol{q} \in \mathcal{Q}$. **Center**: decomposition of the humanoid into left and right halves, whose joint states are represented as $\boldsymbol{q}_\mathrm{l} \in \mathcal{Q}_\mathrm{l}$ and $\boldsymbol{q}_\mathrm{r} \in \mathcal{Q}_\mathrm{r}$, respectively. **Right**: construction of two equivalent single-sided systems by flipping the right half. Their joint states are $\boldsymbol{q}_\mathrm{l} \in \mathcal{Q}_\mathrm{l}$ and $\hat{\mathcal{M}}_\mathrm{q}^{-1}(\boldsymbol{q}_\mathrm{r}) \in \mathcal{Q}_\mathrm{l}$, respectively. They share the same configuration space and can be controlled by the same policy $\hat{\pi}$.

linear maps that combine permutations of coordinates with flipping operations (and, if necessary, constant offsets).

2.1.3 Symmetry Constraints in DRL

To incorporate symmetry into DRL, we model the environment as a symmetric MDP(Zinkevich and Balch, 2001). An MDP $(\mathcal{S}, \mathcal{A}, r, T, p_0)$ is defined as symmetric if there exist symmetry operators $\mathcal{M}_s$ and $\mathcal{M}_a$ under which the reward function, the transition density function, and the initial state distribution are invariant:

$$\begin{aligned} r(\mathcal{M}_s(\boldsymbol{s}), \mathcal{M}_a(\boldsymbol{a})) &= r(\boldsymbol{s}, \boldsymbol{a}), \\ T(\mathcal{M}_s(\boldsymbol{s}') \mid \mathcal{M}_s(\boldsymbol{s}), \mathcal{M}_a(\boldsymbol{a})) &= T(\boldsymbol{s}' \mid \boldsymbol{s}, \boldsymbol{a}), \\ p_0(\mathcal{M}_s(\boldsymbol{s})) &= p_0(\boldsymbol{s}). \end{aligned}$$

Under this symmetry assumption, the optimal policy $\pi^* : \mathcal{S} \times \mathcal{A} \to [0, 1]$ and the optimal value function $V^* : \mathcal{S} \to \mathbb{R}$ are also invariant:

$$\begin{aligned} \pi^*(\mathcal{M}_a(\boldsymbol{a}) \mid \mathcal{M}_s(\boldsymbol{s})) &= \pi^*(\boldsymbol{a} \mid \boldsymbol{s}), \\ V^*(\mathcal{M}_s(\boldsymbol{s})) &= V^*(\boldsymbol{s}). \end{aligned}$$

In DRL for robotic control, a common way to exploit morphological symmetry is to impose invariance on the policy and value function with respect to the symmetry operators:

$$\pi(\mathcal{M}_a(\boldsymbol{a}) \mid \mathcal{M}_s(\boldsymbol{s})) = \pi(\boldsymbol{a} \mid \boldsymbol{s}), \tag{1}$$

$$V(\mathcal{M}_s(\boldsymbol{s})) = V(\boldsymbol{s}) \tag{2}$$

2.2 Policy Symmetrization by Multi Agents

We propose a DRL framework that exploits the morphological symmetry of the robot and redefines the environment as a multi-agent RL system to reduce the complexity of the action space. We define the configuration space of the robot as $\mathcal{Q} \subset \mathbb{R}^{\mathrm{DoF}}$, where DoF denotes the number of degrees of freedom of the system. For a symmetric system, we assume there exists a configuration-level symmetry operator $\mathcal{M}_q : \mathcal{Q} \to \mathcal{Q}$ associated with the generator g of the symmetry group. As in the case of $\mathcal{M}_s$ and $\mathcal{M}_a$, this operator is an involution, i.e., $\mathcal{M}_q^2 = \mathrm{Id}$ and thus $\mathcal{M}_q^{-1} = \mathcal{M}_q$.

Following Abdolhosseini et al. (2019) and Yan et al. (2024), we introduce a decomposition of the joint configuration $\boldsymbol{q} \in \mathcal{Q}$ into left and right joint states $\boldsymbol{q}_{\mathrm{l}} \in \mathcal{Q}_{\mathrm{l}}$ and $\boldsymbol{q}_{\mathrm{r}} \in \mathcal{Q}_{\mathrm{r}}$ with respect to the sagittal plane, as follows (Figure 1):

$$(\boldsymbol{q}_{\mathrm{l}}, \boldsymbol{q}_{\mathrm{r}}) \coloneqq \mathcal{M}_{\mathrm{split}}(\boldsymbol{q}),$$

where the mapping $\mathcal{M}_{\mathrm{split}} : \mathcal{Q} \to \mathcal{Q}_{\mathrm{l}} \times \mathcal{Q}_{\mathrm{r}}$ is a bijection between $\mathcal{Q}$ and the product of left and right configuration spaces $\mathcal{Q}_{\mathrm{l}}$ and $\mathcal{Q}_{\mathrm{r}}$. Joints that lie on the sagittal plane, such as neck or waist joints, are conceptually represented as pairs of virtual left and right joints so that each physical joint is assigned to exactly one of $\boldsymbol{q}_{\mathrm{l}}$ or $\boldsymbol{q}_{\mathrm{r}}$.

For a symmetric system, the following properties hold: (i) $\dim(\mathcal{Q}_{\mathrm{l}}) = \dim(\mathcal{Q}_{\mathrm{r}})$, and (ii) there exists a bijection $\hat{\mathcal{M}}_q : \mathcal{Q}_{\mathrm{l}} \to \mathcal{Q}_{\mathrm{r}}$ such that, for all $\boldsymbol{q} \in \mathcal{Q}$,

$$\begin{aligned} &(\boldsymbol{q}_{\mathrm{l}}, \boldsymbol{q}_{\mathrm{r}}) = \mathcal{M}_{\mathrm{split}}(\boldsymbol{q}), \\ &\mathcal{M}_q(\boldsymbol{q}) = \mathcal{M}_{\mathrm{split}}^{-1}\big(\hat{\mathcal{M}}_q^{-1}(\boldsymbol{q}_{\mathrm{r}}),\, \hat{\mathcal{M}}_q(\boldsymbol{q}_{\mathrm{l}})\big). \end{aligned} \tag{3}$$

Intuitively, $\hat{\mathcal{M}}_q$ maps a left-side configuration to the corresponding right-side configuration, and $\mathcal{M}_q$ acts on the full configuration by swapping the left and right components through $\hat{\mathcal{M}}_q$ and its inverse.

As a concrete example, consider the humanoid robot in Figure 1. For $\mathcal{M}_{\mathrm{split}}$, the full-body joint vector $\boldsymbol{q}$ is divided into two joint vectors, $\boldsymbol{q}_{\mathrm{l}}$ and $\boldsymbol{q}_{\mathrm{r}}$, based on the robot's symmetry plane. Joints located on the symmetry plane are included in both vectors. For $\hat{\mathcal{M}}_q$, the left-side joint vector $\boldsymbol{q}_{\mathrm{l}}$ is reflected across the symmetry plane by flipping the signs of the joints whose rotation axes are parallel to the plane. In the robot shown in Figure 3, this includes, for example, the neck, elbow, hip, and ankle joints.

We now construct a policy that operates only on the left side and is mirrored to create a full-body policy. The block diagram of the policy is shown in Figure 2. In the figure, the entire policy is represented as two policies that share parameters. Let $\hat{\pi}_\theta$ denote the base policy with the left action space $\mathcal{A}_{\mathrm{l}}$, with dimensions that match $\mathcal{Q}_{\mathrm{l}}$. Analogously to the joint configuration, we assume that the full action space $\mathcal{A}$ can be decomposed as

$$\mathcal{M}_{\mathrm{split}} : \mathcal{A} \to \mathcal{A}_{\mathrm{l}} \times \mathcal{A}_{\mathrm{r}},$$

where $\mathcal{A}_{\mathrm{l}}$ and $\mathcal{A}_{\mathrm{r}}$ are the left and right action spaces, respectively, and $\mathcal{M}_{\mathrm{split}}$ is a bijection. With a slight abuse of notation, we use the same symbols $\mathcal{M}_{\mathrm{split}}$ and $\mathcal{M}_{\mathrm{split}}^{-1}$ for the configuration and action spaces.

By reusing the construction of $\hat{\mathcal{M}}_q$, we introduce an action-level symmetry mapping $\hat{\mathcal{M}}_a : \mathcal{A}_{\mathrm{l}} \to \mathcal{A}_{\mathrm{r}}$ between the left and right action spaces. Given a state $\boldsymbol{s}$, we define the full action $\boldsymbol{a} \in \mathcal{A}$ generated by the

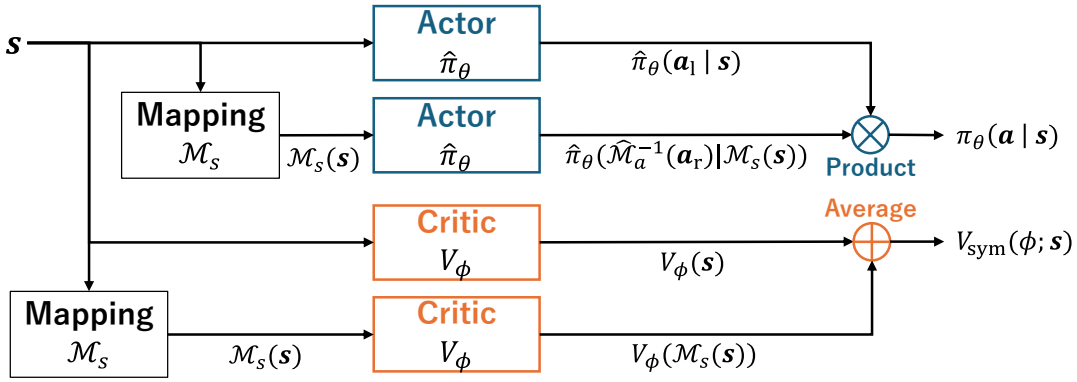

Figure 2. Network architecture of the proposed framework. Modules with the same color (blue or orange) share network parameters.

policy as

$$a \coloneqq \mathcal{M}_{\mathrm{split}}^{-1}(a_\mathrm{l},\, a_\mathrm{r}), \tag{4}$$

$$a_\mathrm{l} \sim \hat{\pi}_\theta(\cdot \mid s), \tag{5}$$

$$a_\mathrm{r} \coloneqq \hat{\mathcal{M}}_a(a'_\mathrm{r}), \quad a'_\mathrm{r} \sim \hat{\pi}_\theta(\cdot \mid \mathcal{M}_s(s)). \tag{6}$$

Here, a_l is a left-side action sampled with state s, a'_r is a left-side action sampled with the mirrored state $\mathcal{M}_s(s)$, and $\hat{\mathcal{M}}_a(a'_\mathrm{r})$ yields the corresponding right-side action. Finally, $\mathcal{M}_{\mathrm{split}}^{-1}$ recombines the left and right components into a full action in $\mathcal{A}$. Within this framework, the operation of a single robot is treated as a multi-agent task involving two agents, each controlling only one half of the body.

As a simple example, consider a locomotion task with the robot shown in Figure 1. Let the state s consist only of the joint positions q, and let the action a be the target joint positions $\dot{q} \in \mathcal{Q}$. In this case, $\mathcal{M}_s(s) = \mathcal{M}_q(q)$. By feeding s and $\mathcal{M}_s(s)$ into the base policy $\hat{\pi}_\theta$, we obtain the side-wise actions a_l and a'_r, which are then combined in the reverse manner of the joint-splitting operation to produce the full-body action a, i.e., the target joint positions $\dot{q}$.

We now show that the resulting full policy π_θ satisfies the invariance condition (1). Let $(a_\mathrm{l}, a_\mathrm{r}) = \mathcal{M}_{\mathrm{split}}(a)$. By construction (4)–(6), the joint density of a under π_θ can be written as

$$\pi_\theta(a \mid s) = \hat{\pi}_\theta(a_\mathrm{l} \mid s)\, \hat{\pi}_\theta\big(\hat{\mathcal{M}}_a^{-1}(a_\mathrm{r}) \mid \mathcal{M}_s(s)\big).$$

Analogously to (3), we define the action-level symmetry operator

$$\mathcal{M}_a(a) \coloneqq \mathcal{M}_{\mathrm{split}}^{-1}\big(\hat{\mathcal{M}}_a^{-1}(a_\mathrm{r}),\, \hat{\mathcal{M}}_a(a_\mathrm{l})\big),$$

so that, if $(\tilde{a}_\mathrm{l}, \tilde{a}_\mathrm{r}) = \mathcal{M}_{\mathrm{split}}(\mathcal{M}_a(a))$, we have $\tilde{a}_\mathrm{l} = \hat{\mathcal{M}}_a^{-1}(a_\mathrm{r})$ and $\tilde{a}_\mathrm{r} = \hat{\mathcal{M}}_a(a_\mathrm{l})$. Applying the same construction (4)–(6) at the mirrored state $\mathcal{M}_s(s)$ yields

$$\begin{aligned}
\pi_\theta(\mathcal{M}_a(a) \mid \mathcal{M}_s(s)) &= \hat{\pi}_\theta(\tilde{a}_\mathrm{l} \mid \mathcal{M}_s(s))\, \hat{\pi}_\theta\big(\hat{\mathcal{M}}_a^{-1}(\tilde{a}_\mathrm{r}) \mid \mathcal{M}_s^2(s)\big) \\
&= \hat{\pi}_\theta\big(\hat{\mathcal{M}}_a^{-1}(a_\mathrm{r}) \mid \mathcal{M}_s(s)\big)\, \hat{\pi}_\theta(a_\mathrm{l} \mid s) \\
&= \pi_\theta(a \mid s),
\end{aligned}$$

where we used the facts that $\hat{\mathcal{M}}_a^{-1} \circ \hat{\mathcal{M}}_a = \mathrm{Id}$ and $\mathcal{M}_s^2 = \mathrm{Id}$. Therefore, the constructed policy π_θ satisfies the invariance condition

$$\pi_\theta(\mathcal{M}_a(\boldsymbol{a}) \mid \mathcal{M}_s(\boldsymbol{s})) = \pi_\theta(\boldsymbol{a} \mid \boldsymbol{s}),$$

and hence is consistent with (1).

2.3 Critic Network Symmetrization

Same as the standard PPO, we train the critic network $V_\phi : \mathcal{S} \to \mathbb{R}$, which is parametrized by ϕ. To maintain invariance as denoted in Equation(2), the value function output is calculated by,

$$V_{\mathrm{sym}}(\phi; \boldsymbol{s}) = \frac{V_\phi(\boldsymbol{s}) + V_\phi(\mathcal{M}_{\mathrm{s}}(\boldsymbol{s}))}{2}.$$

2.4 Coupled Objective Function

In addition to the proposed symmetric policy architecture, we further modify the objective of the underlying DRL algorithm. By working in a smaller action space $\mathcal{A}_{\mathrm{l}}$ for the base policy (and generating right-side actions by symmetry), the complexity of the policy network can be reduced. However, the left and right action spaces are closely related because they jointly control a single robot; therefore, policy updates must be performed with caution. In Yan et al. (2024), the authors directly apply a multi-agent RL algorithm based on PPO (Schulman et al., 2017). In a simple MARL implementation (e.g., MAPPO (Yu et al., 2022)), the actor objective is given by

$$J_{\mathrm{MARL}}(\theta) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a}_{\mathrm{l}},\boldsymbol{a}_{\mathrm{r}})\sim\mathcal{D}}\left[\frac{1}{2}\sum_{i\in\{\mathrm{l},\mathrm{r}\}} \min\Big(\rho_i(\theta;\boldsymbol{s},\boldsymbol{a}_i)\, A_i(\boldsymbol{s},\boldsymbol{a}_i), \mathrm{clip}\big(\rho_i(\theta;\boldsymbol{s},\boldsymbol{a}_i), 1-\epsilon, 1+\epsilon\big)\, A_i(\boldsymbol{s},\boldsymbol{a}_i)\Big)\right].$$

The importance-sampling ratios for the left and right sides are defined as

$$\rho_{\mathrm{l}}(\theta;\boldsymbol{s},\boldsymbol{a}_{\mathrm{l}}) = \frac{\hat{\pi}_\theta(\boldsymbol{a}_{\mathrm{l}} \mid \boldsymbol{s})}{\hat{\pi}_{\mathrm{old}}(\boldsymbol{a}_{\mathrm{l}} \mid \boldsymbol{s})},$$
$$\rho_{\mathrm{r}}(\theta;\boldsymbol{s},\boldsymbol{a}_{\mathrm{r}}) = \frac{\hat{\pi}_\theta\big(\hat{\mathcal{M}}_{\mathrm{a}}^{-1}(\boldsymbol{a}_{\mathrm{r}}) \mid \mathcal{M}_s(\boldsymbol{s})\big)}{\hat{\pi}_{\mathrm{old}}\big(\hat{\mathcal{M}}_{\mathrm{a}}^{-1}(\boldsymbol{a}_{\mathrm{r}}) \mid \mathcal{M}_s(\boldsymbol{s})\big)},$$

where $\hat{\pi}_{\mathrm{old}}$ denotes the policy before the update, $\mathcal{D}$ denotes the data distribution collected by $\hat{\pi}_{\mathrm{old}}$, and $A_i(\boldsymbol{s},\boldsymbol{a})$ is the advantage function computed using the GAE estimator (Schulman et al., 2016).

However, this objective treats the left and right action components as if they were controlled by two independent agents that share parameters but are updated using separate importance-sampling ratios. Since the full policy π_θ is constructed as the left policy with inputs of the original state and the mirrored state, this decoupled treatment can lead to inconsistent updates between the two sides, even though they jointly control a single robot. In particular, the clipping is applied independently to ρ_{l} and ρ_{r}, which may over- or under-penalize updates for configurations where the symmetry plays an essential role.

To better respect the symmetric structure of the policy, we consider the full policy

$$\pi_\theta(\boldsymbol{a} \mid \boldsymbol{s}) = \hat{\pi}_\theta(\boldsymbol{a}_{\mathrm{l}} \mid \boldsymbol{s})\, \hat{\pi}_\theta\big(\hat{\mathcal{M}}_{\mathrm{a}}^{-1}(\boldsymbol{a}_{\mathrm{r}}) \mid \mathcal{M}_s(\boldsymbol{s})\big),$$

and define a single coupled importance-sampling ratio

$$\rho_{\mathrm{sym}}(\theta; \boldsymbol{s}, \boldsymbol{a}) \coloneqq \frac{\pi_\theta(\boldsymbol{a} \mid \boldsymbol{s})}{\pi_{\mathrm{old}}(\boldsymbol{a} \mid \boldsymbol{s})} = \rho_{\mathrm{l}}(\theta; \boldsymbol{s}, \boldsymbol{a}_{\mathrm{l}})\, \rho_{\mathrm{r}}(\theta; \boldsymbol{s}, \boldsymbol{a}_{\mathrm{r}}),$$

where π_{old} is defined similarly using $\hat{\pi}_{\mathrm{old}}$. Since our method assumes a symmetric MDP environment with reward and value function symmetry, the advantage function is invariant and can be defined as $A(\boldsymbol{s}, \boldsymbol{a}) = A_{\mathrm{l}}(\boldsymbol{s}, \boldsymbol{a}_{\mathrm{l}}) = A_{\mathrm{r}}(\boldsymbol{s}, \boldsymbol{a}_{\mathrm{r}})$. We then replace the MAPPO-style objective by the following symmetric PPO objective:

$$J_{\mathrm{sym}}(\theta) = \mathbb{E}_{(\boldsymbol{s},\boldsymbol{a})\sim\mathcal{D}}\Big[\min\big(\rho_{\mathrm{sym}}(\theta; \boldsymbol{s}, \boldsymbol{a})\, A(\boldsymbol{s}, \boldsymbol{a}), \mathrm{clip}(\rho_{\mathrm{sym}}(\theta; \boldsymbol{s}, \boldsymbol{a}), 1-\epsilon, 1+\epsilon)\, A(\boldsymbol{s}, \boldsymbol{a})\big)\Big]. \tag{7}$$

This objective performs clipping based on the ratio of the coupled symmetric policy rather than on the per-side ratios separately, and thus updates the left and right actions in a joint manner.

By definition, the symmetric ratio is invariant under the symmetry operators: for any $(\boldsymbol{s}, \boldsymbol{a})$,

$$\rho_{\mathrm{sym}}(\theta; \mathcal{M}_s(\boldsymbol{s}), \mathcal{M}_a(\boldsymbol{a})) = \rho_{\mathrm{sym}}(\theta; \boldsymbol{s}, \boldsymbol{a}),$$

because π_θ itself satisfies $\pi_\theta(\mathcal{M}_a(\boldsymbol{a}) \mid \mathcal{M}_s(\boldsymbol{s})) = \pi_\theta(\boldsymbol{a} \mid \boldsymbol{s})$. Consequently, the coupled PPO objective (7) is consistent with the invariance condition (1) and encourages policy updates that preserve the underlying morphological symmetry of the robot.

2.5 Observation Symmetrization

Symmetric Environment

We now describe how we define the symmetry operator on observations, $\mathcal{M}_s : \mathcal{S} \to \mathcal{S}$, for all states $\boldsymbol{s} \in \mathcal{S}$. In DRL for robotic control, the observation is typically composed of intrinsic and extrinsic components. The intrinsic component contains proprioceptive information of the robot, such as joint and link states. The extrinsic component encodes information of the external environment, such as the states of manipulated objects. For locomotion tasks, most of the observations are intrinsic, whereas manipulation tasks require rich extrinsic information.

We represent the observation as a tuple

$$\boldsymbol{s} \coloneqq (\boldsymbol{s}_{\mathrm{intr}}, \boldsymbol{s}_{\mathrm{extr}}, 1, 0),$$

where $\boldsymbol{s}_{\mathrm{intr}}$ and $\boldsymbol{s}_{\mathrm{extr}}$ denote the intrinsic and extrinsic components, respectively. As described above, $\mathcal{M}_s$ is implemented as a combination of permutation and value-flipping (with constant offsets) operators. Let P denote a permutation operator and F a value-flipping operator. We then define the symmetrized observation as

$$\mathcal{M}_s(\boldsymbol{s}) = \big(F_{\mathrm{intr}} \circ P_{\mathrm{intr}}(\boldsymbol{s}_{\mathrm{intr}}),\ F_{\mathrm{extr}} \circ P_{\mathrm{extr}}(\boldsymbol{s}_{\mathrm{extr}}),\ 0, 1\big), \tag{8}$$

where P_{intr} and P_{extr} are the permutation operators for intrinsic and extrinsic components, and F_{intr} and F_{extr} are the corresponding value-flipping operators.

The last two entries form a one-hot vector that is flipped under $\mathcal{M}_s$, i.e., $(1, 0) \mapsto (0, 1)$. This prevents the observation from becoming neutral in the sense that we avoid configurations for which $\mathcal{M}_s(\boldsymbol{s}) = \boldsymbol{s}$ and, consequently, $\mathcal{M}_a(\boldsymbol{a}) = \boldsymbol{a}$. Existing work on locomotion, such as Abdolhosseini et al. (2019); Mittal et al.

(2024), uses a phase signal Liu et al. (2016) to characterize the gait cycle of each leg and to avoid such neutral configurations. However, phase signals rely on the periodicity of the motion, whereas our method targets more general tasks, including manipulation, which do not necessarily exhibit periodic behavior. We therefore use the one-hot indicator instead of a phase signal.

Asymmetric Environment

Although the proposed method assumes that the underlying environment is symmetric, it is rare in practice for the dataset collected by an agent to be perfectly symmetric. In this setting, we cannot directly use the observation symmetry operator $\mathcal{M}_s$ defined in (8), because, for asymmetric tasks, there is no well-defined permutation P_{extr} on the extrinsic states that pairs objects on the left with corresponding objects on the right. Instead, we introduce a modified (asymmetric) version of the operator, which we denote by the same symbol when the context is clear:

$$\mathcal{M}_s(\boldsymbol{s}) = \big(F_{\mathrm{intr}} \circ P_{\mathrm{intr}}(\boldsymbol{s}_{\mathrm{intr}}),\ F_{\mathrm{extr}}(\boldsymbol{s}_{\mathrm{extr}}),\ 0, 1\big). \tag{9}$$

This operator transforms the intrinsic state in the same way as in the symmetric case (permutation plus value flipping), while for the extrinsic state it only flips values without permuting object identities. A permutation operator for the intrinsic state can be defined from the robot's morphological symmetry, but, in general, no such canonical permutation exists for the extrinsic state. For instance, in manipulation tasks involving multiple non-identical objects, permuting the entries of the extrinsic observation would distort the semantic correspondence among the objects. Therefore, for the extrinsic component we apply only value flipping, which mirrors the object poses while preserving object identities. As a result, $\mathcal{M}_s$ is no longer an exact symmetry of the MDP, but it still provides a meaningful way to construct asymmetric counterparts of observed states.

3 EXPERIMENTAL SETUP

We perform experimental validation on several tasks in simulation and deploy the learned policies to real-world robots through sim-to-real transfer.

3.1 Comparative Methods and Training Environments

We compared the five methods as follows: PPO, Augmentation, Equivariant MLP, Multi-Agent, and Ours. Implementation is based on the PPO algorithm from RL-Games (Makoviichuk and Makoviychuk, 2021).

- **PPO:** The standard PPO policy without symmetric constraints.
- **Augmentation (Aug):** We augment the collected sample using symmetric operations $\mathcal{M}_s$ and $\mathcal{M}_a$, and append it to the dataset. We referred to Mittal et al. (2024) for implementation.
- **Equivariant MLP (EMLP):** We explicitly constrain the MLP network of both the policy and the critic to be equivariant to the prescribed symmetry. Concretely, EMLP represents the weights in each hidden layer as a linear combination of basis matrices that are equivariant under the symmetry transformation and treats the corresponding coefficients as trainable parameters. Our implementation follows Su et al. (2024).
- **Multi-Agent (MARL):** We control the robot using two agents that share the same policy network in a multi-agent reinforcement learning manner. Implementation is described in section 2.2 and 2.3.

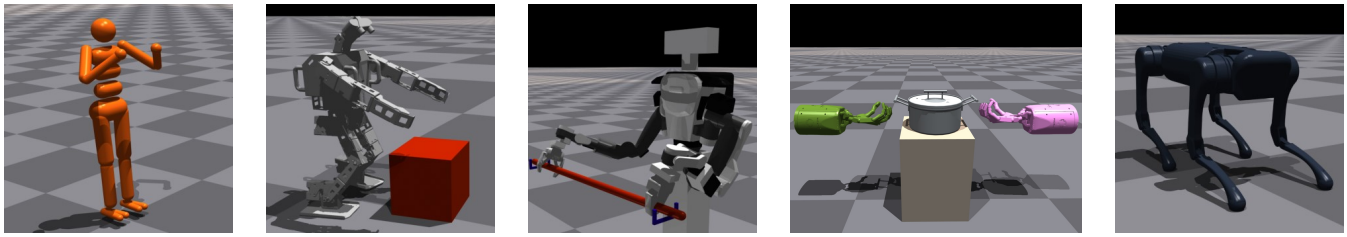

Figure 3. The robots used in experiments: **(A)** CG Humanoid, **(B)** Full-Body Humanoid Robot, **(C)** Upper-Body Humanoid Robot, **(D)** Bi-DexHands, and **(E)** Quadruped Robot.

- **Multi-Agent with Coupled loss (Ours):** Our proposed method with the coupled objective is detailed in section 2.4.

We trained all the agents in environments in the Isaac Gym simulator (Makoviychuk et al., 2021). For each task, we conducted five trials with different seed values.

3.2 Robotic Platforms and Tasks

We use five morphologically symmetric robotic platforms: a simulated (CG) full-body humanoid, a real full-body humanoid robot, an upper-body humanoid robot, a pair of five-fingered robotic hands, and a quadruped robot. Figure 3 shows the robots used in the experiments. Implementation details are provided in the appendix.

- **CG Humanoid**
  For the CG humanoid, we employ the humanoid environment from IsaacGymEnvs.[1]
  - **Locomotion:** The humanoid runs under torque control in a fixed direction.
  - **Backflip:** The humanoid performs a backflip in place.
- **Full-Body Humanoid Robot**
  We use a ROBOTIS OP3 full-body humanoid robot with a height of 0.5 m. All joints are driven by DYNAMIXEL servomotors controlled in position mode through the DYNAMIXEL Python API.[2]
  - **Locomotion:** The robot walks in a commanded direction. The direction command is resampled at random time intervals.
  - **Pickup Box:** The robot picks up a box placed 0.5–1.0 m in front of the robot. The box is a cube with an edge length of 0.15 m.
- **Upper-Body Humanoid Robot**
  We use an upper-body humanoid robot, HIRO (KAWADA Robotics), equipped with two 1-DoF end-effectors (ROBOTIS Hand RH-P12-RN).
  - **Grasp Rod:** The robot grasps a 1 m rod placed on a holder with both hands.
- **Bi-DexHands Benchmark**
  We employ the Bi-DexHands benchmark (Chen et al., 2024), which contains various tasks conducted by a pair of robot dexterous hands with five fingers. We modify the environments to enforce a symmetric configuration. More specifically, since the original setup consists of two right-sided hands, we mirror

[1] https://github.com/isaac-sim/IsaacGymEnvs

[2] https://emanual.robotis.com/docs/en/software/dynamixel/dynamixel_sdk/overview/

the MJCF and mesh to create a left-sided hand. From the benchmark, we chose the following tasks which can be formulated as symmetric MDPs.

  - **Door Open Inward:** The hands open double-hinged doors that can only be pulled inward.
  - **Lift Underarm:** The hands grasp a pot by its handle with both hands and lift it upward.
  - **Two Catch Underarm:** Each hand initially holds a ball, and the hands swap the balls by throwing them to the opposite side.

- **Quadruped Robot**
  The quadruped robot we use is a Unitree A1.
  - **Locomotion:** The locomotion direction is specified by linear and angular velocity commands.
  - **Handstand:** The robot performs a handstand by balancing on its forelegs in place.

3.3 Asymmetric Tasks

Additionally, we evaluate our method on asymmetric tasks, which do not satisfy the symmetric MDP assumption defined in Section 2.1.3. In these experiments, we investigate how asymmetric data skew affects performance by training a symmetric policy for asymmetric environments. It should be noted that we do not evaluate the augmentation-based method (**Aug**), since the policy network is asymmetric (as in standard PPO) and the symmetry-augmented samples are not compatible with the true environment dynamics.

For the asymmetric experiments, we use the environments from the Bi-DexHands benchmark (Chen et al., 2024).

- **Block Stack:** The hands stack two blocks on a table to form a tower. The desired stacking order is specified in the reward function.
- **Grasp and Place:** The hands place a small cube into a cup on the table. The left hand is closer to the cup, and the right hand is closer to the box.
- **Scissors:** The hands open a pair of scissors on the table.
- **Pen:** The hands remove the cap from a pen on the table. The left hand is closer to the cap, and the right hand is closer to the pen.
- **Catch Underarm:** The left hand throws a ball to the right hand, which catches it. This task is a one-handed version of Two Catch Underarm.
- **Catch Abreast:** Similar to Catch, except that the hands are positioned in parallel.

3.4 Evaluation Metrics

We evaluated the methods with two evaluation metrics: *highest reward* and *steps-to-threshold*.

- **Highest Reward:** The maximum episode return during training. We trained five agents for each condition and evaluated the performance using the mean value. The range of returns among tasks differs widely, so the value is normalized by the value of the baseline PPO.
- **Steps-to-Threshold:** The number of training iterations required for the agent to achieve a given reward threshold. The threshold is set to 50%, 75%, 90%, and 100% of the highest reward achieved by the baseline PPO. The iteration count is normalized by the maximum number of training iterations for each task.

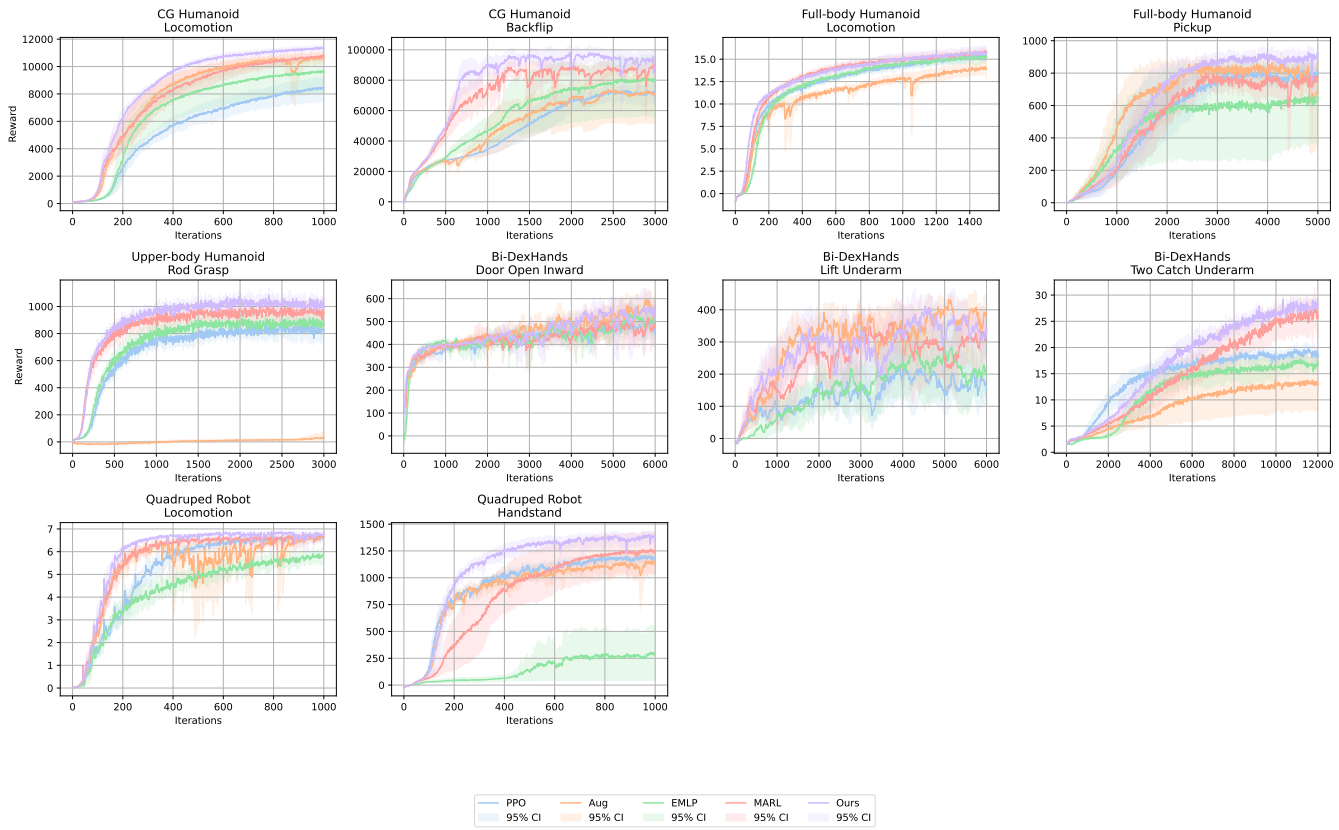

Figure 4. Reward curves for symmetric tasks across five runs with different random seeds. The light-colored areas indicate 95% confidence intervals.

4 RESULTS

4.1 Evaluation on Symmetric Tasks

First, we evaluate the performance of the symmetric-aware policies for the symmetric tasks. Figure 4 shows the reward curves for symmetric tasks averaged over five trials for each condition.

4.1.1 Highest Reward Comparison

We evaluate the reward acquisition performance of five methods: **PPO**, **Aug**, **EMLP**, **MARL**, and **Ours**. Figure 5 and the left column of Table 3 report the highest returns averaged over five trials for each condition; Figure 5 shows 95% confidence intervals, whereas Table 3 reports the results as mean $\pm$ standard deviation. For nine out of the ten tasks, except for the locomotion task of the full-body humanoid robot, the proposed method achieves the highest normalized reward among all methods. The augmentation-based method (**Aug**) and the equivariant MLP-based method (**EMLP**) sometimes outperform **PPO**, but they suffer from substantial performance drops on specific tasks (e.g., *Grasp Rod* by the upper-body humanoid robot for **Aug**, and *Handstand* with the quadruped robot for **EMLP**). The multi-agent method (**MARL**), which shares the same network architecture as the proposed method but uses a different objective function, consistently improves over **PPO** similar to **Aug** and **EMLP**, but without such pronounced degradation on particular tasks.

Another distinctive feature of the proposed method is its lower variance across training runs with different random seeds. This is evident from the standard deviations reported in Table 3, which summarizes the

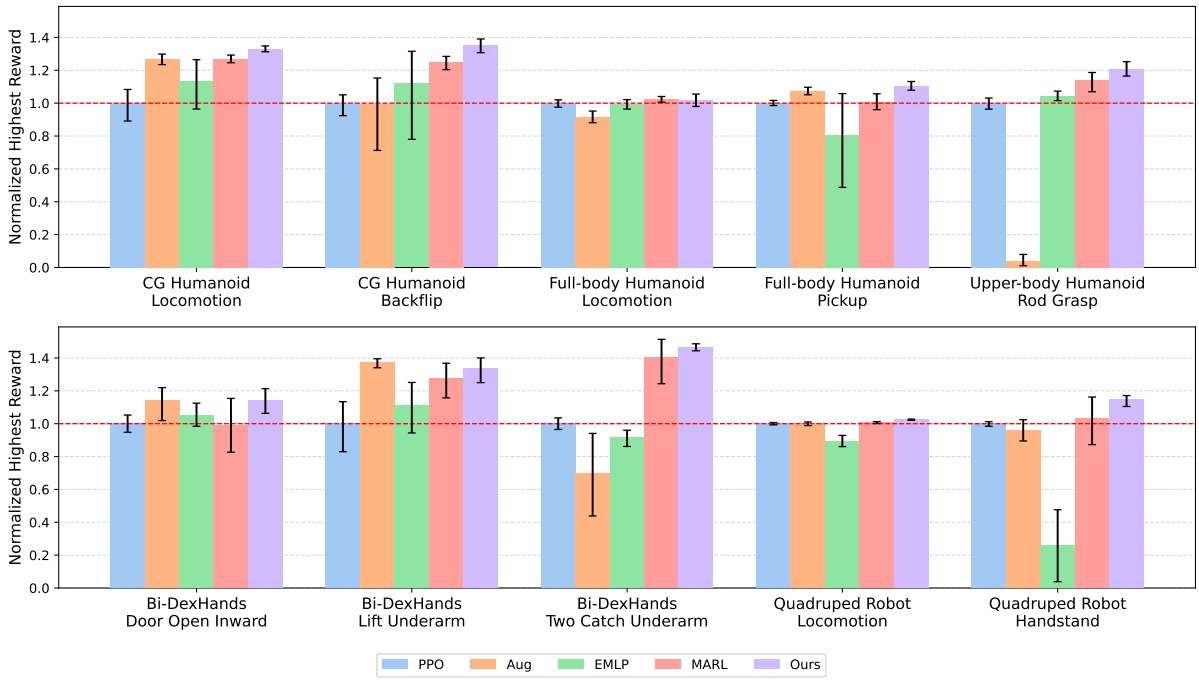

Figure 5. Highest reward comparison for symmetric tasks. The values are normalized by the value of PPO. The error bars indicate 95% confidence intervals.

results as mean ± standard deviation:: **Ours** achieves the smallest standard deviation in 5 out of 10 tasks, and its standard deviation remains below 0.1 in all tasks.

In other words, the performance of these baselines depends strongly on stochastic factors such as network initialization and random state transitions, whereas our method is comparatively less sensitive to such sources of randomness. Collectively, these results indicate that the proposed framework is the most robust to the inevitable stochasticity in DRL among the methods considered.

4.1.2 Steps-to-Threshold Comparison

Table 3 presents the normalized number of iterations required to reach given fractions of the final PPO return on symmetric tasks. Our method (**Ours**) reaches 100% of the PPO return fastest on seven out of ten tasks. If we also include the second-best method, **Ours** ranks at best or second-best on nine out of ten tasks.

PPO generally acquires rewards more slowly than the symmetry-based methods. However, in some tasks, such as *Two Catch Underarm* with Bi-DexHands and *Handstand* with the quadruped robot, **PPO** exhibits a characteristic pattern: it is initially the fastest method, but its learning slows down, and it is eventually overtaken by the symmetry-based approaches. By inspecting the learned behaviors, we found that this pattern arises because PPO converges to a suboptimal strategy in which only one of the two objects is transferred, yielding rapid initial reward gains but preventing further improvement toward the solution where both objects are swapped.

Aug shows highly task-dependent, "peaky" performance. It achieves reward acquisition speeds comparable to **Ours** on *Locomotion* with the CG humanoid, *Pickup Box* with the full-body humanoid, and *Door Open Inward* and *Lift Underarm* with Bi-DexHands, but lags behind on the other tasks.

EMLP consistently struggles to match the other methods: it never ranks in the top two on any given task.

Table 3. Highest reward and steps-to-threshold comparison for symmetric tasks across five runs with different random seeds (mean ± std). The best value is shown in **red**, and the second-best value is shown in **blue**.

Task	Method	Highest Reward (↑)	Steps-to-Threshold (↓) 50%	75%	90%	100%
CG Humanoid Locomotion	PPO	1±0.123	0.276±0.036	0.517±0.124	0.737±0.191	0.864±0.142
	Aug	**1.27±0.0402**	0.201±0.0466	**0.255±0.0524**	**0.319±0.0688**	**0.385±0.0942**
	EMLP	1.136±0.195	0.231±0.059	0.351±0.201	0.484±0.294	0.577±0.261
	MARL	1.268±0.0291	**0.187±0.0331**	0.27±0.0723	0.358±0.117	0.423±0.131
	Ours	**1.331±0.0218**	**0.156±0.022**	**0.201±0.0221**	**0.257±0.0156**	**0.311±0.0191**
CG Humanoid Backflip	PPO	1±0.0858	0.386±0.0839	0.55±0.0919	0.691±0.181	0.817±0.131
	Aug	0.993±0.312	0.455±0.312	0.535±0.275	0.571±0.256	0.671±0.193
	EMLP	1.12±0.374	0.392±0.345	0.528±0.286	0.574±0.263	0.609±0.255
	MARL	**1.248±0.0511**	**0.133±0.0251**	**0.224±0.0581**	**0.265±0.0723**	**0.284±0.0788**
	Ours	**1.351±0.0564**	**0.125±0.0093**	**0.19±0.01**	**0.222±0.0386**	**0.235±0.0458**
Full-body Humanoid Locomotion	PPO	1±0.0278	0.0881±0.0044	0.233±0.033	0.58±0.0706	0.943±0.075
	Aug	0.917±0.0446	0.101±0.0044	0.387±0.0195	0.905±0.126	1±0
	EMLP	0.993±0.037	0.105±0.0087	0.226±0.0202	0.525±0.0735	0.943±0.0817
	MARL	**1.023±0.0214**	**0.0824±0.0084**	**0.168±0.0188**	**0.379±0.0459**	**0.825±0.159**
	Ours	**1.02±0.0479**	**0.0661±0.0039**	**0.159±0.0023**	**0.419±0.0821**	**0.807±0.193**
Full-body Humanoid Pickup Box	PPO	1±0.0196	0.337±0.106	0.401±0.114	0.503±0.119	0.903±0.159
	Aug	**1.074±0.0297**	**0.227±0.123**	**0.256±0.12**	**0.299±0.106**	**0.443±0.151**
	EMLP	0.808±0.365	0.368±0.355	0.557±0.407	0.603±0.373	0.753±0.34
	MARL	1.006±0.0627	0.301±0.0848	0.397±0.112	0.489±0.0924	0.847±0.22
	Ours	**1.104±0.0344**	**0.289±0.076**	**0.325±0.0834**	**0.367±0.11**	**0.454±0.13**
Upper-body Humanoid Grasp Rod	PPO	1±0.0454	0.129±0.0122	0.226±0.0175	0.434±0.0845	0.872±0.128
	Aug	0.0373±0.0465	1±0	1±0	1±0	1±0
	EMLP	1.041±0.0368	0.118±0.0088	0.21±0.0367	0.293±0.0576	0.646±0.231
	MARL	**1.14±0.0772**	**0.0637±0.005**	**0.106±0.0203**	**0.161±0.0263**	**0.263±0.124**
	Ours	**1.209±0.0564**	**0.0636±0.0027**	**0.0989±0.0089**	**0.143±0.0099**	**0.211±0.0252**
Bi-DexHands Door Open Inward	PPO	1±0.0686	0.0178±0.0054	0.321±0.147	0.66±0.274	0.957±0.0716
	Aug	**1.138±0.13**	0.0284±0.0081	0.172±0.0282	**0.565±0.261**	**0.645±0.222**
	EMLP	1.053±0.0865	0.0252±0.0035	**0.132±0.0601**	0.571±0.225	0.828±0.156
	MARL	0.99±0.207	**0.0149±0.0016**	0.28±0.226	0.683±0.312	0.713±0.294
	Ours	**1.143±0.0954**	**0.0175±0.0065**	**0.145±0.0647**	**0.49±0.0796**	**0.672±0.108**
Bi-DexHands Lift Underarm	PPO	1±0.252	0.151±0.0488	0.451±0.312	0.708±0.277	0.8±0.28
	Aug	**1.423±0.0558**	0.131±0.0808	**0.146±0.0861**	**0.172±0.0954**	**0.228±0.126**
	EMLP	1.269±0.0855	0.238±0.0506	0.362±0.0347	0.484±0.133	0.537±0.143
	MARL	1.402±0.0883	**0.0871±0.007**	0.178±0.0826	**0.189±0.0887**	**0.199±0.0867**
	Ours	**1.463±0.025**	**0.114±0.0552**	**0.162±0.0536**	0.215±0.0699	0.276±0.0347
Bi-DexHands Two Catch Underarm	PPO	1±0.0455	**0.18±0.0442**	**0.306±0.0823**	**0.558±0.0932**	0.933±0.0986
	Aug	0.697±0.331	0.598±0.282	0.788±0.291	0.86±0.195	0.987±0.03
	EMLP	0.915±0.0638	0.286±0.0565	0.521±0.216	0.844±0.214	1±0
	MARL	**1.405±0.183**	0.309±0.0719	0.443±0.108	0.559±0.141	**0.638±0.149**
	Ours	**1.465±0.0283**	**0.255±0.0265**	**0.345±0.0467**	**0.416±0.0676**	**0.478±0.0679**
Quadruped Robot Locomotion	PPO	1±0.0093	0.172±0.0129	0.264±0.0387	0.405±0.0512	0.947±0.0738
	Aug	1±0.0149	0.114±0.0079	**0.159±0.0066**	0.245±0.0307	0.932±0.0858
	EMLP	0.895±0.0435	0.195±0.0555	0.451±0.157	0.92±0.115	1±0
	MARL	**1.007±0.0076**	**0.112±0.0101**	0.159±0.0093	**0.235±0.0146**	**0.812±0.18**
	Ours	**1.025±0.0048**	**0.105±0.0078**	**0.14±0.0145**	**0.188±0.0118**	**0.415±0.047**
Quadruped Robot Handstand	PPO	1±0.0178	**0.139±0.0101**	**0.268±0.0397**	**0.471±0.0571**	0.943±0.0733
	Aug	0.96±0.0825	0.148±0.0046	0.291±0.0705	0.743±0.265	0.946±0.0825
	EMLP	0.257±0.288	0.873±0.19	1±0	1±0	1±0
	MARL	**1.032±0.194**	0.281±0.0929	0.518±0.308	0.6±0.283	**0.698±0.28**
	Ours	**1.148±0.048**	**0.145±0.0229**	**0.192±0.0174**	**0.273±0.0383**	**0.393±0.116**

MARL generally learns faster than **PPO** but falls short of **Ours** on most tasks. The only exception is *Lift Underarm* with Bi-DexHands, where **MARL** reaches the target reward faster than **Ours**, although its highest reward remains lower.

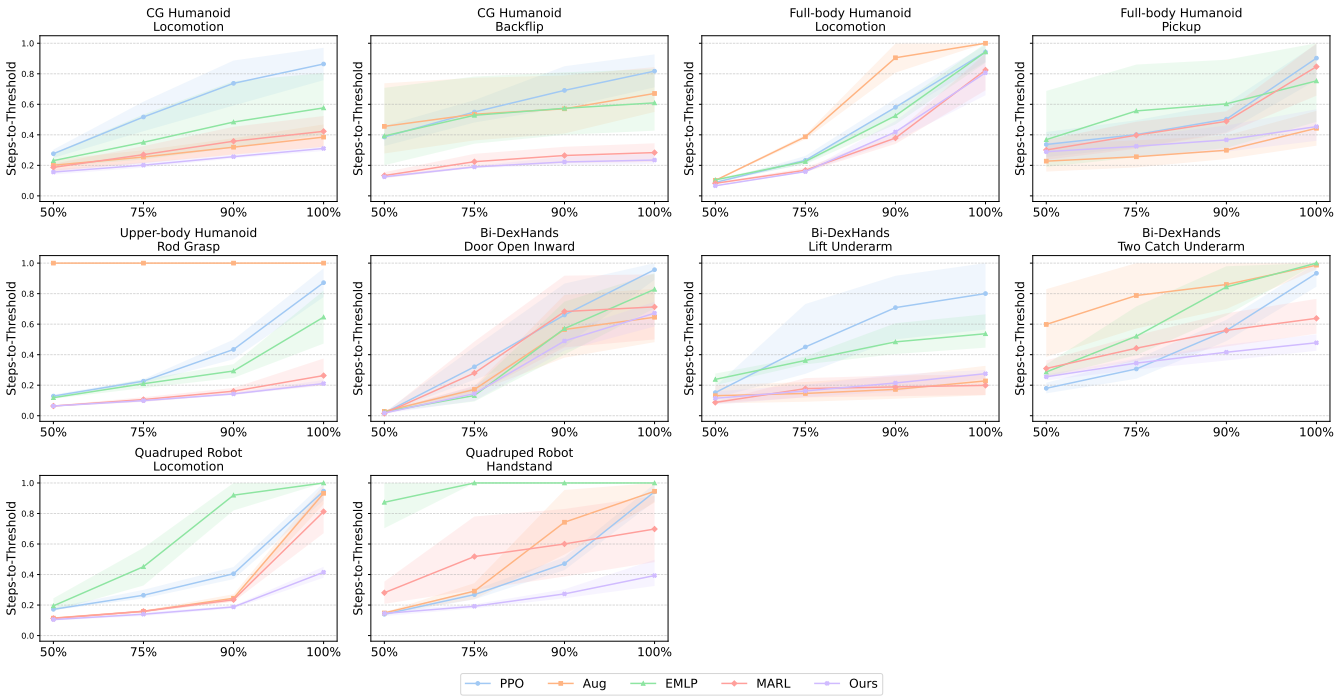

Figure 6. Steps-to-Threshold for symmetric tasks across five runs with different random seeds. Lower values indicate faster reward acquisition. The error bars indicate 95% confidence intervals.

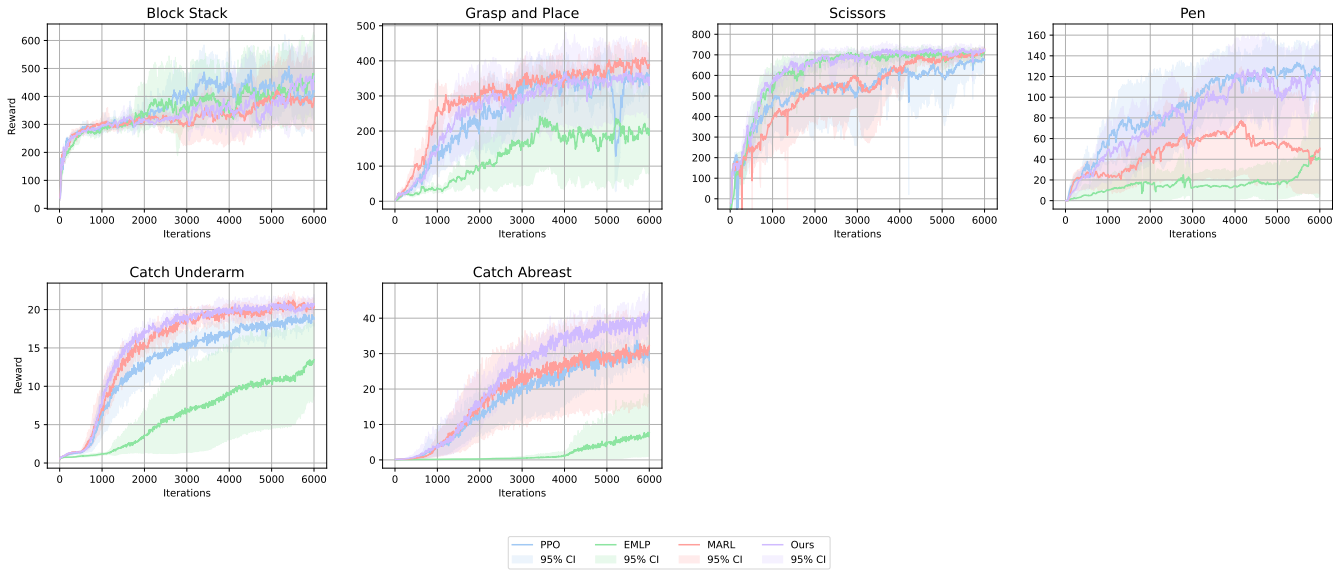

Figure 7. Reward curves for asymmetric tasks across five runs with different random seeds. The light-colored areas indicate 95% confidence intervals.

4.2 Evaluation on Asymmetric Tasks

Next, we evaluate the performance of the symmetry-aware methods for the asymmetric tasks. We compare four methods—**PPO**, **EMLP**, **MARL**, and **Ours**—excluding **Aug**, since augmented data is not compatible with the asymmetric tasks. Figure 7 shows the reward curves for the asymmetric tasks averaged over five trials for each condition.

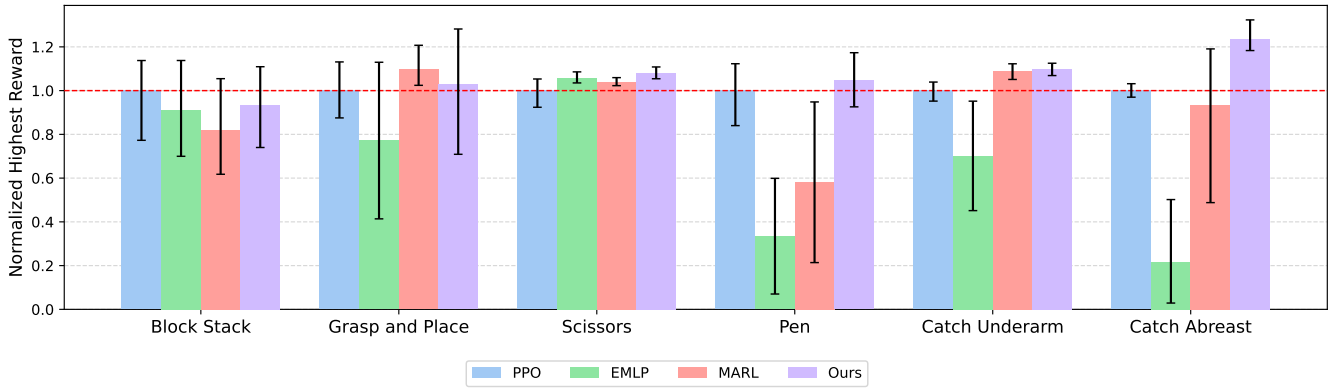

Figure 8. Highest reward comparison for asymmetric tasks across five runs with different random seeds. The values are normalized by the value of PPO. The error bars indicate 95% confidence intervals.

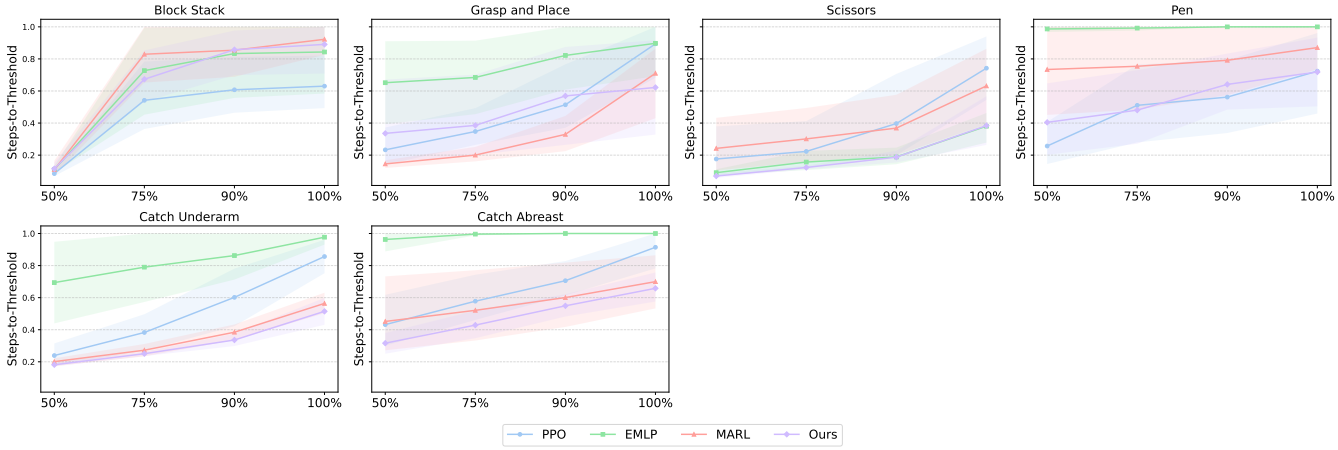

Figure 9. Steps-to-Threshold for asymmetric tasks across five runs with different random seeds. Lower values indicate faster reward acquisition. The error bars indicate 95% confidence intervals.

4.2.1 Highest Reward Comparison

Figure 8 and the left column of Table 4 report the highest returns averaged over five trials for each condition; Figure 8 shows 95% confidence intervals, whereas Table 4 reports the results as mean ± standard deviation. Our method outperforms the other baselines on four out of the six tasks; it underperforms **PPO** on *Block Stack* and underperforms **MARL** on *Grasp and Place*. In contrast, **EMLP** achieves lower rewards than **PPO** on five out of the six tasks and, in particular, exhibits a significant performance drop on *Pen* and *Catch Abreast*. **MARL** attains higher returns than **PPO** on some tasks, but its performance shows high variance across different random seeds.

4.2.2 Comparison of Steps-to-Threshold

Figure 9 and the right four columns of Table 4 depict the steps-to-threshold for the asymmetric tasks; Figure 9 shows 95% confidence intervals, whereas Table 4 reports the results as mean ± standard deviation. Although it is not as significant compared to the symmetric tasks, our method shows relatively fast reward acquisition in comparison to the other methods.

Table 4. Training speed and highest reward comparison for asymmetric tasks across five runs with different random seeds (mean ± std). The best value is shown in **red**, and the second-best value is shown in **blue**.

Task	Method	Highest Reward (↑)	Steps-to-Threshold (↓) 50%	75%	90%	100%
Block Stack	PPO	**1±0.252**	**0.0849±0.0177**	**0.542±0.271**	**0.608±0.23**	**0.63±0.219**
	EMLP	0.911±0.281	0.11±0.0267	0.727±0.36	**0.833±0.3**	**0.843±0.281**
	MARL	0.821±0.278	**0.109±0.0636**	0.829±0.234	0.854±0.203	0.922±0.112
	Ours	**0.934±0.236**	0.115±0.0447	**0.673±0.204**	0.858±0.186	0.891±0.197
Grasp and Place	PPO	1±0.151	**0.233±0.151**	**0.347±0.158**	**0.514±0.274**	0.894±0.237
	EMLP	0.772±0.481	0.652±0.329	0.684±0.294	0.822±0.252	0.897±0.23
	MARL	**1.097±0.125**	**0.145±0.0314**	**0.2±0.0595**	**0.33±0.142**	**0.71±0.308**
	Ours	**1.028±0.365**	0.336±0.372	0.385±0.347	0.569±0.397	**0.622±0.368**
Scissors	PPO	1±0.0855	0.176±0.227	0.223±0.209	0.397±0.345	0.742±0.252
	EMLP	**1.059±0.0323**	**0.0907±0.024**	**0.157±0.0784**	**0.189±0.0668**	**0.38±0.116**
	MARL	1.041±0.0237	0.242±0.218	0.301±0.221	0.369±0.235	0.632±0.303
	Ours	**1.08±0.0364**	**0.0698±0.0177**	**0.123±0.0137**	**0.187±0.047**	**0.384±0.207**
Pen	PPO	**1±0.178**	**0.257±0.17**	**0.511±0.327**	**0.562±0.312**	**0.724±0.322**
	EMLP	0.333±0.346	0.987±0.0204	0.992±0.0174	1±0	1±0
	MARL	0.581±0.469	0.734±0.363	0.754±0.34	0.791±0.286	0.871±0.178
	Ours	**1.047±0.159**	**0.404±0.297**	**0.481±0.3**	**0.641±0.232**	**0.719±0.273**
Catch Underarm	PPO	1±0.0554	0.239±0.0884	0.383±0.142	0.602±0.231	0.856±0.138
	EMLP	0.702±0.32	0.694±0.32	0.79±0.288	0.862±0.19	0.977±0.0516
	MARL	**1.088±0.0445**	**0.201±0.033**	**0.272±0.0483**	**0.384±0.063**	**0.564±0.0836**
	Ours	**1.098±0.0367**	**0.181±0.0175**	**0.251±0.0177**	**0.336±0.0511**	**0.515±0.109**
Catch Abreast	PPO	**1±0.0401**	**0.432±0.203**	0.578±0.183	0.706±0.142	0.914±0.143
	EMLP	0.215±0.321	0.963±0.0831	0.996±0.0088	1±0	1±0
	MARL	0.935±0.495	0.452±0.314	**0.521±0.288**	**0.601±0.257**	**0.7±0.207**
	Ours	**1.236±0.0979**	**0.316±0.0836**	**0.429±0.109**	**0.549±0.0886**	**0.658±0.113**

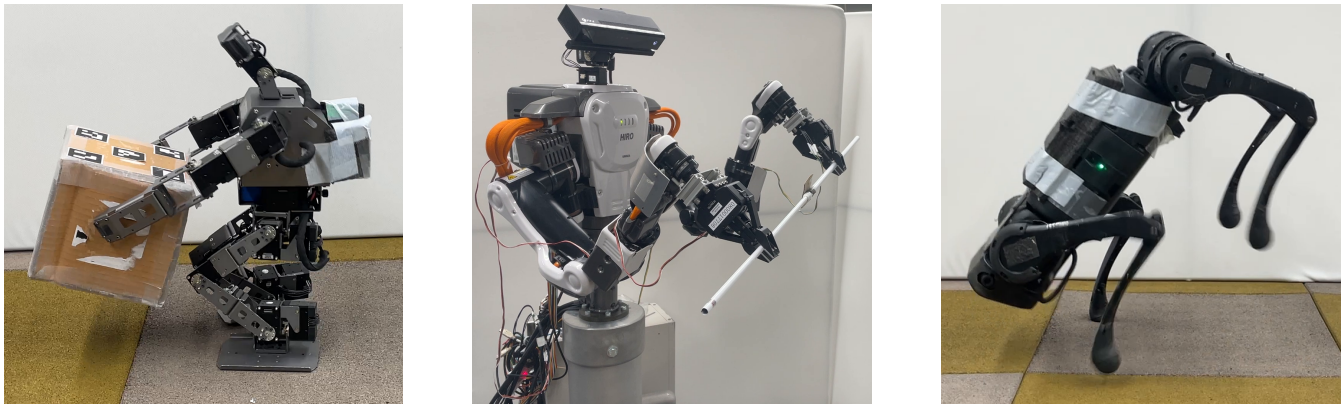

Figure 10. Sim-to-Real Transfer. **Left**: *Pickup Box* by full-body humanoid robot (Robotis OP3), **Center**: *Grasp Rod* by upper-body humanoid robot (HIRO), and **Right**: *Handstand* by quadruped robot (Unitree A1).

4.3 Sim-to-Real Transfer

As shown in Fig. 10, we deployed the trained policies for *Pickup Box* on a full-body humanoid robot, *Rod Grasp* on an upper-body humanoid robot, and *Handstand* on a quadruped robot. For *Pickup Box* and *Rod Grasp*, AR markers were used to estimate the pose of the target object. Videos of the real-world deployment are provided in the supplemental materials.

Table 5. Domain randomization parameters for OP3

Term	Range
Joint stiffness	[11, 21.5]
Joint damping	[0.2, 0.5]
Body friction	[0.5, 1.0]
Body restitution	[0.0, 0.3]
System delay	[25 ms, 100 ms]
Joint angle offset	$[-2.9^\circ, 2.9^\circ]$
IMU rotational offset	$[0^\circ, 2^\circ]$
Additional weight	[0 kg, 0.5 kg]
Additional weight pos	[-7.5 cm, 7.5 cm]
Random push force	[2 N, 5 N]
Random push force interval	[3 s, 5 s]

Table 6. Domain randomization parameters for HIRO

Term	Range
Target positional noise	[-0.025 m, 0.025 m]
Target rotational noise	$[-5^\circ, 5^\circ]$

Table 7. Domain randomization parameters for A1

Term	Range
Joint stiffness	[24, 36]
Joint damping	[0.6, 0.9]
Body friction	[0.4, 2.0]
Body restitution	[0.0, 0.4]
System delay	[10 ms, 40 ms]
Additional link masses:	
-Base	[-1 kg, 1 kg]
-Hip	[0, 0.2 kg]
-Thigh	[-0.1 kg, 0.1 kg]
-Calf	[-0.03 kg, 0.03 kg]
-Foot	[-0.01 kg, 0.01 kg]
Link COM offset	[-0.02 m, 0.02 m]
Random push force	[0, 30 N]
Random push force interval	[5 s, 10 s]

To transfer the trained policies to the real robots, we applied domain randomization by adding noise to the observations, actions, and environment dynamics during training. Tables 5, 6, and 7 summarize the domain randomization parameters used for each robot.

We evaluated the real-world success rates of the three tasks. For *Pickup Box*, a trial was considered successful if the robot lifted the box off the ground and held it steadily. In this task, we evaluated two conditions in which the robot was initially positioned 0.5 m and 1.0 m away from the target box. For *Rod Grasp*, success was defined as grasping the rod with both hands and lifting it. For *Handstand*, success was defined as maintaining a foreleg stand for at least five seconds. Table 8 presents the real-world success rates over 10 trials for the three tasks. These results suggest that the proposed method learns policies that transfer robustly to real-world settings.

Table 8. Number of successes over 10 trials for sim-to-real transfer.

Task	PPO	Aug	EMLP	MARL	Ours
Full-body Humanoid *Pickup Box*-0.5 m	4/10	7/10	3/10	6/10	8/10
Full-body Humanoid *Pickup Box*-1.0 m	0/10	5/10	1/10	7/10	7/10
Upper-body Humanoid *Rod Grasp*	7/10	0/10	8/10	9/10	9/10
Quadruped Robot *Handstand*	10/10	9/10	6/10	3/10	10/10

5 DISCUSSION

5.1 Performance on Symmetric Tasks

We observe that **Ours**, which combines a multi-agent policy network with a modified symmetric objective, outperforms the other methods on most tasks in terms of both reward acquisition and learning speed. Although the other symmetry-based methods (**Aug**, **EMLP**, and **MARL**) sometimes achieve higher or faster reward acquisition than **PPO**, they also suffer from substantially degraded performance on specific tasks.

In supervised learning, exploiting morphological symmetry—through data augmentation or symmetric architectural constraints—often leads to consistent performance gains. In DRL, however, such approaches are not necessarily effective to the same extent. A key difference is the highly stochastic nature of DRL: the training dataset (i.e., the experience collected during exploration) is not guaranteed to be symmetrically distributed, even if the underlying environment is perfectly symmetric. Symmetric constraints on the policy network can encourage more balanced behavior, but they cannot fully compensate for asymmetries introduced by stochastic initializations and transitions. For example, in locomotion tasks, the commanded walking direction is typically sampled randomly, and in manipulation tasks, the initial positions of target objects are randomized. These sources of randomness inevitably induce skew in the collected experience, which can conflict with rigid symmetry constraints and, in turn, degrade the performance of symmetry-based methods.

Our method imposes symmetry, but in a different way: it enforces structure at the level of policy construction and uses a joint symmetric objective derived from a MARL-style formulation, where imbalance in the experiences of different "agents" (left and right sides) is expected rather than ignored. This design allows the policy to exploit symmetry when present, while remaining comparatively robust to skewed data.

In addition, the comparison between **Ours** and **MARL** highlights the effect of the coupled objective. Across the benchmark tasks, **Ours** improves both the highest reward and the steps-to-threshold in most cases. As shown in Table 3, **Ours** achieves a higher reward in 9 out of 10 symmetric tasks, except for *Locomotion* by the full-body humanoid, and a smaller steps-to-threshold at 100% in 9 out of 10 symmetric tasks, except for *Two Catch Underarm* by Bi-DexHands. This trend is further supported by the bootstrap 95% confidence intervals shown in Figures 5 and 6. In particular, **Ours** shows non-overlapping confidence intervals for the highest reward in *CG Humanoid Locomotion*, *CG Humanoid Backflip*, *Pickup Box* by the full-body humanoid, and *Locomotion* by the quadruped robot, suggesting that the improvement is unlikely to be explained solely by seed-level variation. Similarly, for the steps-to-threshold at 100%, non-overlapping confidence intervals are observed in *Pickup Box* by the full-body humanoid, as well as in *Locomotion* and *Handstand* by the quadruped robot.

While splitting the action space into two half-body policies reduces the complexity of the control problem, the **MARL** objective remains defined at the level of the individual sides. Accordingly, the clipping

operation is applied independently to the left and right importance-sampling ratios. In standard PPO, clipping constrains the policy update by restricting the ratio to $[1-\epsilon,\ 1+\epsilon]$. In contrast, when the two sides are clipped independently, the effective ratio of the full policy may deviate more strongly; for example, if both per-side ratios change in the same direction, the combined ratio can reach $(1-\epsilon)^2$ or $(1+\epsilon)^2$. This suggests that the decoupled objective may permit more aggressive updates than intended for the full policy.

By clipping the coupled ratio of the full symmetric policy, **Ours** ties the updates of the two sides together and better matches the structure of the policy itself. We hypothesize that this is one of the main reasons why **Ours** consistently outperforms **MARL**.

5.2 Robustness to Asymmetric Environments

Ours remains competitive with **PPO** even on asymmetric tasks, despite being designed under the assumption of a symmetric MDP. In contrast, **EMLP** is worse than **PPO** on five out of the six asymmetric tasks, and the **MARL** variant typically lies between **EMLP** and **Ours**. This behavior is particularly noteworthy given the MARL-style design of **Ours**. As defined in Equation (9), our observation mapping enforces symmetry only on the intrinsic state, while no permutation-based symmetry constraint is imposed on the extrinsic state.

Although the advantage of the proposed method on asymmetric tasks is smaller than that observed on symmetric tasks, **Ours** does not degrade substantially and maintains performance at a level comparable to or better than the baseline **PPO**, whereas **EMLP** and **MARL** suffer large performance drops on some tasks. While **Ours** underperforms **PPO** on *Block Stack*, the gap is generally modest and is often accompanied by overlapping confidence intervals, making the difference inconclusive given the current number of random seeds. These results suggest that the coupled objective primarily improves robustness against asymmetric environments rather than merely exploiting symmetry-specific structure. In practical settings, it is rare for the environment and task to strictly satisfy the symmetric MDP assumptions. Our findings indicate that the proposed method can perform well across a wide range of tasks, as long as the robot itself exhibits morphological symmetry, even when the surrounding environment is asymmetric.

6 CONCLUSION

In this work, we proposed a symmetry-assisted DRL framework for morphologically symmetric robots that exploits left–right reflection symmetry at both the policy and training objective levels. We modeled the environment as a symmetric MDP equipped with symmetry operators on the state, action, and configuration spaces, and constructed a full-body policy from a single base policy acting on one side of the robot. By mirroring observations and actions through these operators, our method guarantees an equivariant policy structure while reducing the effective action dimension and retaining the simplicity of standard MLP architectures.

To align the optimization procedure with this structural symmetry, we further introduced a symmetric PPO objective based on a coupled importance-sampling ratio for the full policy. Unlike MAPPO-style MARL formulations, which share parameters but treat the left and right sides as separate agents with per-side clipping, our objective updates both sides in a coordinated fashion that is consistent with the underlying symmetry. We compared this framework with four baselines—PPO, data augmentation, EMLP, and a MARL variant —on ten symmetric and six asymmetric tasks across five robotic platforms, including simulated and real humanoid robots, a quadruped, and dexterous multi-fingered hands.

There are several directions for future work. First, extending the framework to more general symmetry groups beyond simple reflections (e.g., cyclic or permutation groups for multi-legged or modular robots) could further broaden its applicability. Second, combining our symmetric PPO objective with richer value-function architectures or model-based components may improve sample efficiency in more complex tasks. Second, while our focus in this work is on symmetry-assisted DRL, combining the proposed objective with richer value-function architectures or model-based components may further improve sample efficiency in more complex tasks. In particular, for contact-rich robotic systems where accurate analytical contact-dynamics models are available, model-based trajectory optimization methods such as DDP(Singh et al., 2023) offer a complementary route to improving sample efficiency. Likewise, for long-horizon or multi-stage robotic tasks, hierarchical model-based structures such as primitive-based planning(Sun et al., 2025) may provide useful inductive bias for improving data efficiency and execution reliability. Finally, a deeper theoretical analysis of the interplay among symmetry, data skew, and optimization dynamics could lead to principled criteria for choosing among architectural, objective-level, and hybrid symmetry-exploiting methods in practical robotic systems.

7 APPENDIX

In the appendix section, we detail the implementation details for the tasks in the experiment, such as state space, action space, their corresponding symmetry operators, and hyperparameters of DRL.

7.1 Table Conventions

Before presenting the robot and task details, we first explain how to interpret the tables. We provide the joint list and intrinsic state variables for each robot, along with the extrinsic state variables for each task.

Each joint list includes the joint ID, joint name, side label, the joint-wise flipping operator F_{joint}, and the joint-wise permutation operator P_{joint}. The side label indicates whether the joint belongs to the left or right joint-configuration space, $\mathcal{Q}_l$ or $\mathcal{Q}_r$. For F_{joint}, a value of -1 indicates sign inversion of the corresponding joint state, while a value of 1 indicates that the state remains unchanged. P_{joint} specifies the target joint ID under the permutation.

Each intrinsic and extrinsic state table includes the environment-related state vectors, the flipping operator $F_{\mathrm{intr/extr}}$, and the permutation operator $P_{\mathrm{intr/extr}}$. In $F_{\mathrm{intr/extr}}$, $\mathrm{flip}(\cdot)$ denotes the sign inversion of the state-vector elements at the specified indices. In $P_{\mathrm{intr/extr}}$, $\mathrm{to}(\cdot)$ denotes the swapping of the corresponding vectors.

The action symmetry operators $\mathcal{M}_a$ are defined based on the joint lists, whereas the state symmetry operators $\mathcal{M}_s$ are defined based on the intrinsic and extrinsic state tables. For MARL-based methods, the split operator $\mathcal{M}_{\mathrm{split}}$ is determined by the side labels in the joint lists.

7.2 CG Humanoid

7.2.1 Intrinsic state space

CG humanoid has 21 joints as shown in Table 9. The intrinsic state of the humanoid is shown in Table 10

7.2.2 Action Space and Actuation

Let $\mathbf{a} \in \mathbb{R}^{21}$ denote the action, where each element corresponds to one actuated DoF of the humanoid. The action is interpreted as a normalized actuation command and converted into a joint force command

Table 9. Joint list and symmetric operators of CG Humanoid.

ID	Name	Side	F_{joint}	P_{joint}
0	Abdomen Z	L+R	-1	0
1	Abdomen Y	L+R	1	1
2	Abdomen X	L+R	-1	2
3	Right Hip X	R	1	9
4	Right Hip Z	R	1	10
5	Right Hip Y	R	1	11
6	Right Knee	R	1	12
7	Right Ankle Y	R	1	13
8	Right Ankle X	R	-1	14
9	Left Hip X	L	1	3
10	Left Hip Z	L	1	4
11	Left Hip Y	L	1	5
12	Left Knee	L	1	6
13	Left Ankle Y	L	1	7
14	Left Ankle X	L	-1	8
15	Right Shoulder1	R	1	18
16	Right Shoulder2	R	1	19
17	Right Elbow	R	1	20
18	Left Shoulder1	L	1	15
19	Left Shoulder2	L	1	16
20	Left Elbow	L	1	17
Total	21			

Table 10. Intrinsic states and symmetry operators of CG Humanoid

Symbol	Dimension	Description	F_{intr}	P_{intr}
h_t	1	Torso height	1	-
$\mathbf{v}_{\text{loc}}$	3	Torso linear velocity	$\text{flip}(1)$	-
$\boldsymbol{\omega}_{\text{loc}}$	3	Torso angular velocity	$\text{flip}(0,1)$	-
$\mathbf{R}_{\text{root}}$	4	Torso rotation quaternion	$\text{flip}(1,3)$	-
$\mathbf{q}$	21	Joint positions	F_{joint}	P_{joint}
$\dot{\mathbf{q}}$	21	Joint velocities	F_{joint}	P_{joint}
$\boldsymbol{\tau}$	21	Joint torque	F_{joint}	P_{joint}
$\mathbf{f}_{\text{L}}$	6	Foot force/torque sensor	$\text{flip}(1,3,5)$	$\text{to}(\mathbf{f}_{\text{R}})$
$\mathbf{f}_{\text{R}}$	6	Foot force/torque sensor	$\text{flip}(1,3,5)$	$\text{to}(\mathbf{f}_{\text{L}})$
$\mathbf{a}_{\text{prev}}$	21	Previous action	F_{joint}	P_{joint}
Total	106			

through

$$\boldsymbol{\tau} = \tau^{\max}\mathbf{a}, \tag{10}$$

where $\tau^{\max}$ is the actuator effort limit obtained from the humanoid asset. The robot is actuated by the calculated joint force at 60 Hz.

7.2.3 Task Details

***Locomotion*:** Table 11 shows the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Locomotion* task with CG Humanoid. The target direction is fixed as the forward vector (the $+x$ direction), and the agent learns to run toward it.

***Backflip*:** Table 12 shows the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Backflip* task with CG Humanoid. The backflip motion is divided into five phases:

Table 11. Extrinsic states and symmetry operators of *Locomotion* by CG Humanoid

Symbol	Dimension	Description	F_{extr}	P_{extr}
θ	1	Relative angle to the target direction	$\mathrm{flip}(0)$	-
u	1	Torso upper vector projection	-	-
d	1	Torso forward vector projection	-	-
Total	3			

Table 12. Extrinsic states and symmetry operators of *Backflip* by CG Humanoid

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{o}$	5	One-hot vector for backflip phase	-	-
Total	5			

standing, rotated by $90°$, rotated by $180°$, rotated by $270°$, and landing. The extrinsic state is represented by a five-dimensional one-hot vector $\mathbf{o}$ that indicates the current phase of the backflip.

7.2.4 Training Parameters

Table 13 summarizes the training hyperparameters used for the CG Humanoid tasks.

Table 13. Hyperparameters in training of CG Humanoid

Parameter	Value
Actor hidden layers	[400, 200, 100]
Critic hidden layer	[400, 200, 100]
Activation	ELU
Clip range	0.2
Discount factor	0.99
GAE discount factor	0.95
Desired KL divergence	0.008
Learning rate	5×10^{-4}
Number of Environments	4096
Steps per iteration	32
Training iterations	
-*Locomotion*	1000
-*Backflip*	3000

7.3 Full-body Humanoid

7.3.1 Intrinsic state space

The full-body humanoid robot (OP3, ROBOTIS Corp.) has 20 joints, as shown in Table 14. The intrinsic state is shown in Table 15. The intrinsic states from the previous five time steps are stacked and fed into the policy.

7.3.2 Action Space and Actuation

Let $\mathbf{a} \in \mathbb{R}^{20}$ denote the action vector, whose elements correspond to the actuated DoFs of the robot. The action is interpreted as a normalized target joint position. The commanded target joint position is then

Table 14. Joint list of Full-body Humanoid

ID	Name	Side	F_{joint}	P_{joint}
0	Head Pan	L+R	-1	0
1	Head Tilt	L+R	1	1
2	Left Hip Yaw	L	-1	11
3	Left Hip Roll	L	-1	12
4	Left Hip Pitch	L	-1	13
5	Left Knee	L	-1	14
6	Left Ankle Pitch	L	-1	15
7	Left Ankle Roll	L	-1	16
8	Left Shoulder Pitch	L	-1	17
9	Left Shoulder Roll	L	-1	18
10	Left Elbow	L	-1	19
11	Right Hip Yaw	R	-1	2
12	Right Hip Roll	R	-1	3
13	Right Hip Pitch	R	-1	4
14	Right Knee	R	-1	5
15	Right Ankle Pitch	R	-1	6
16	Right Ankle Roll	R	-1	7
17	Right Shoulder Pitch	R	-1	8
18	Right Shoulder Roll	R	-1	9
19	Right Elbow	R	-1	10
Total	20			

Table 15. Intrinsic states and symmetry operators of Full-body Humanoid

Symbol	Dimension	Description	F_{intr}	P_{intr}
$\boldsymbol{\omega}_{\text{loc}}$	3	Local angular velocity	$\text{flip}(0, 2)$	-
$\mathbf{g}_{\text{loc}}$	3	Projected gravity direction	$\text{flip}(1)$	-
$\mathbf{q}$	18	Joint positions	F_{joint}	P_{joint}
$\tilde{\mathbf{q}}_{\text{prev}}$	18	Previous joint target positions	F_{joint}	P_{joint}
Total	42			

Table 16. Extrinsic states and symmetry operators of *Locomotion* by Full-body Humanoid

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{d}_{\text{loc}}$	3	Projected target heading direction	$\text{flip}(1)$	-
Total	3			

computed as

$$\tilde{\mathbf{q}} = 0.8\tilde{\mathbf{q}}_{\text{prev}} + 0.2\text{unnorm}(\mathbf{a}), \tag{11}$$

where $\tilde{\mathbf{q}}$ is the current target joint position, $\tilde{\mathbf{q}}_{\text{prev}}$ is the target joint position at the previous time step, and $\text{unnorm}(\cdot)$ maps the normalized action to the joint limits. Each joint is controlled by a PD controller at 40 Hz.

7.3.3 Task Details

***Locomotion*:** Table 16 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Locomotion* task with full-body humanoid robot. The target heading direction is randomly sampled during training and provided by a higher-level controller during inference, and the agent is trained to run toward it.

The extrinsic states from the previous five time steps are stacked and fed into the policy.

Table 17. Extrinsic states and symmetry operators of *Pickup Box* by Full-body Humanoid

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{p}_{\mathrm{box}}$	3	Position of the box	$\mathrm{flip}(1)$	-
$\mathbf{u}_{\mathrm{box}}$	3	Up vector (+z) of the box	$\mathrm{flip}(1)$	-
$\mathbf{f}_{\mathrm{box}}$	3	Forward vector (+x) of the box	$\mathrm{flip}(1)$	-
Total	9			

***Pickup Box*:** Table 17 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Pickup Box* task with the full-body humanoid robot. The extrinsic state is defined by the position of the target box and its orientation, where the orientation is represented by the up and forward vectors. The extrinsic states from the previous five time steps are stacked and provided as input to the policy.

7.3.4 Training Parameters

Table 18 summarizes the training hyperparameters used for the Full-body Humanoid tasks.

Table 18. Hyperparameters in training of Full-body Humanoid

Parameter	Value
Actor hidden layers	[256, 128, 64]
Critic hidden layer	[256, 128, 64]
Activation	ELU
Clip range	0.2
Discount factor	0.99
GAE discount factor	0.95
Desired KL divergence	0.008
Learning rate	5×10^{-4}
Number of Environments	4096
Steps per iteration	32
Training iterations	
-Locomotion	1500
-Pickup Box	5000

7.4 Upper-body Humanoid

7.4.1 Intrinsic State Space

Upper-body Humanoid Robot (HIRO, Kawada Robotics Corp. and RH-P12-RN, Robotis Corp.) has 15 joints, as shown in Table 19. The intrinsic state is shown in Table 20.

7.4.2 Action Space and Actuation

Let $\mathbf{a} \in \mathbb{R}^{15}$ denote the action vector, whose elements correspond to the actuated DoFs of the robot. The action is interpreted as the deviation of joint position from the previous time step. The commanded target joint position is then computed as

$$\tilde{\mathbf{q}} = \mathbf{q}_{\mathrm{prev}} + \alpha \mathbf{a}, \tag{12}$$

where $\tilde{\mathbf{q}}$ is the current target joint position, $\mathbf{q}_{\mathrm{prev}}$ is the joint position at the previous time step, and α is coefficient of joint deviation. Each joint is controlled by a PD controller at 10 Hz.

Table 19. Joint list of Upper-body Humanoid

ID	Name	Side	F_{joint}	P_{joint}
0	Chest Joint	L+R	-1	0
1	Left Arm Joint0	L	-1	8
2	Left Arm Joint1	L	1	9
3	Left Arm Joint2	L	1	10
4	Left Arm Joint3	L	-1	11
5	Left Arm Joint4	L	1	12
6	Left Arm Joint5	L	-1	13
7	Left Hand	L	1	14
8	Rarm Joint0	R	-1	1
9	Rarm Joint1	R	1	2
10	Rarm Joint2	R	1	3
11	Rarm Joint3	R	-1	4
12	Rarm Joint4	R	1	5
13	Rarm Joint5	R	-1	6
14	Right Hand	R	1	7
Total	15			

Table 20. Intrinsic states and symmetry operators of Upper-body Humanoid

Symbol	Dimension	Description	F_{intr}	P_{intr}
$\mathbf{q}$	15	Joint positions	F_{joint}	P_{joint}
Total	15			

Table 21. Extrinsic states and symmetry operators of *Grasp Rod* by Upper-body Humanoid

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{p}_{\text{rod}}$	3	Position of the rod center	$\text{flip}(1)$	-
$\mathbf{a}_{\text{rod}}$	3	Axis direction of the rod	$\text{flip}(1)$	-
Total	6			

7.4.3 Task Details

***Grasp Rod*:** Table 21 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Grasp Rod* task with the upper-body humanoid robot. The extrinsic state is defined by the center position and axis direction of the rod.

7.4.4 Training Parameters

Table 22 summarizes the training hyperparameters used for the Upper-body Humanoid tasks.

7.5 Bi-DexHands

7.5.1 Intrinsic state space

Bi-DexHands has 40 joints and 12 DoFs of base poses as shown in Table 23. For convenience, we define the symmetric operators both for joints (F_{joint} and P_{joint}) and base poses (F_{base} and P_{base}) and represent the symmetric operator for entire DoFs as $F_{\text{joint}} \circ F_{\text{base}}$ and $P_{\text{joint}} \circ P_{\text{base}}$. The intrinsic state is shown in Table 24.

Table 22. Hyperparameters in training of Upper-body Humanoid

Parameter	Value
Actor hidden layers	[512, 256, 64]
Critic hidden layer	[512, 256, 64]
Activation	ELU
Clip range	0.2
Discount factor	0.99
GAE discount factor	0.95
Desired KL divergence	0.008
Learning rate	5×10^{-4}
Number of Environments	4096
Steps per iteration	8
Training iterations	3000

7.5.2 Action Space and Actuation

Let $\mathbf{a} \in \mathbb{R}^{52}$ denote the action vector, whose elements correspond to the actuated joints and the base wrenches of the robot. The subvector $\mathbf{a}_{0:39}$ is interpreted as a normalized target joint position, and the commanded target joint position is given by

$$\tilde{\mathbf{q}} = \mathrm{unnorm}(\mathbf{a}_{0:39}), \tag{13}$$

where $\tilde{\mathbf{q}}$ denotes the current target joint position, and $\mathrm{unnorm}(\cdot)$ maps the normalized action to the joint limits. Each joint is controlled by a PD controller at 60 Hz.

The subvector $\mathbf{a}_{40:52}$ represents the forces and torques applied to the base links of the left and right hands:

$$\boldsymbol{f}_{\mathrm{L}} = f_{\max}\mathbf{a}_{40:42}, \tag{14}$$

$$\boldsymbol{\tau}_{\mathrm{L}} = \tau_{\max}\mathbf{a}_{43:45}, \tag{15}$$

$$\boldsymbol{f}_{\mathrm{R}} = f_{\max}\mathbf{a}_{46:48}, \tag{16}$$

$$\boldsymbol{\tau}_{\mathrm{R}} = \tau_{\max}\mathbf{a}_{49:52}. \tag{17}$$

Here, $\boldsymbol{f}_{\mathrm{L}}, \boldsymbol{f}_{\mathrm{R}} \in \mathbb{R}^3$ are the forces applied to the left and right hand bases, respectively, $\boldsymbol{\tau}_{\mathrm{L}}, \boldsymbol{\tau}_{\mathrm{R}} \in \mathbb{R}^3$ are the corresponding torques, and $f_{\max}, \tau_{\max} \in \mathbb{R}$ are the force and torque limits.

7.5.3 Task Details

***Door Open Inward*:** Table 25 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Door Open Inward* task with Bi-DexHands. The extrinsic state contains the position of the handle of doors.

***Lift Underarm*:** Table 26 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Lift Underarm* task with Bi-DexHands. The extrinsic state is defined by the position, orientation, linear velocity, and angular velocity of the pot, together with the positions of the pot handles.

***Two Catch Underarm*:**

Table 27 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Two Catch Underarm* task with Bi-DexHands. The extrinsic state consists of

the positions, orientations, linear velocities, and angular velocities of the two balls, as well as their goal positions and orientations.

7.5.4 Training Parameters

Table 28 summarizes the training hyperparameters used for the Bi-DexHands tasks.

7.6 Quadruped Robot

7.6.1 Intrinsic state space

Quadruped Robot (A1, Unitree Corp.) has 12 joints, as shown in Table 29. The intrinsic state is shown in Table 30.

7.6.2 Action Space and Actuation

Let $\mathbf{a} \in \mathbb{R}^{12}$ denote the action vector, whose elements correspond to the actuated DoFs of the robot. The action is interpreted as a target joint position. The commanded target joint position is then computed as

$$\tilde{\mathbf{q}} = \mathbf{q}_{\text{def}} + \alpha \mathbf{a}, \tag{18}$$

where $\tilde{\mathbf{q}}$ is the current target joint position, $\mathbf{q}_{\text{def}}$ is the default joint position, and α is action coefficient. Each joint is controlled by a PD controller at 50 Hz.

7.6.3 Task Details

***Locomotion*:** Table 31 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Locomotion* task with the quadruped robot. The target linear and angular velocities are randomly sampled during training and provided by a higher-level controller during inference, and the agent is trained to move according to these commands.

***Handstand*:** No extrinsic state is used for the *Handstand* task with the quadruped robot. The agent is trained to maintain a stable handstand while minimizing body sway.

7.6.4 Training Parameters

Table 32 summarizes the training hyperparameters used for the Quadruped robot tasks.

7.7 Asymmetric Tasks

For asymmetric tasks, we use Bi-DexHands environments. The intrinsic states and action space are detailed in Section 7.5.1 and 7.5.2.

7.7.1 *Block Stack* Details

Table 33 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Block Stack* task.

7.7.2 *Grasp and Place* Details

Table 34 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Grasp and Place* task.

7.7.3 *Scissors* Details

Table 35 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Scissors* task.

7.7.4 *Pen* Details

Table 36 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Pen* task.

7.7.5 *Catch Underarm* and *Catch Abreast* Details

Table 37 presents the extrinsic states and their corresponding symmetry operators used for state symmetrization in the *Catch Underarm* and *Catch Abreast* task.

7.7.6 Training Parameters

Table 38 summarizes the training hyperparameters used for the asymmetric tasks.

DATA AVAILABILITY STATEMENT

The training code of this article will be made available online by the authors.

AUTHOR CONTRIBUTIONS

RH: original draft, Methodology, Simulation implementation, Training, Result Analysis, and Sim-to-Real implementation for the full-body humanoid and upper-body humanoid. YL: Sim-to-Real implementation for quadruped robot. MH, YS, JT and KI: Rewiew and Editing. TO: Supervision, Equipment acquisition, Review and Editing.

FUNDING

This work was funded by JSPS KAKENHI Grant Numbers JP24K21173 and JP24H00351. This work was also supported by the UTokyo-SMBC Forest GX project.

REFERENCES

Abdolhosseini, F., Ling, H. Y., Xie, Z., Peng, X. B., and van de Panne, M. (2019). On Learning Symmetric Locomotion. In Proceedings of the 12th ACM SIGGRAPH Conference on Motion, Interaction and Games (New York, NY, USA: Association for Computing Machinery), MIG '19, 1–10

Apraez, D. O., Turrisi, G., Kostic, V., Martin, M., Agudo, A., Moreno-Noguer, F., et al. (2025). Morphological symmetries in robotics. The International Journal of Robotics Research , 02783649241282422

Bao, K., Li, C., As, Y., Krause, A., and Hutter, M. (2025). Toward task generalization via memory augmentation in meta-reinforcement learning. arXiv preprint arXiv:2502.01521

Barakat, A., Fatkhullin, I., and He, N. (2023). Reinforcement learning with general utilities: Simpler variance reduction and large state-action space. In International Conference on Machine Learning (PMLR), 1753–1800

Chen, Y., Geng, Y., Zhong, F., Ji, J., Jiang, J., Lu, Z., et al. (2024). Bi-dexhands: Towards human-level bimanual dexterous manipulation. IEEE Transactions on Pattern Analysis and Machine Intelligence 46, 2804–2818. doi:10.1109/TPAMI.2023.3339515

Dong, H., Zhang, J., Wang, T., and Zhang, C. (2023). Symmetry-aware robot design with structured subgroups. In International Conference on Machine Learning (PMLR), 8334–8355

Finzi, M., Welling, M., and Wilson, A. G. (2021). A practical method for constructing equivariant multilayer perceptrons for arbitrary matrix groups. In International conference on machine learning (PMLR), 3318–3328

Ghaffari, M., Zhang, R., Zhu, M., Lin, C. E., Lin, T.-Y., Teng, S., et al. (2022). Progress in symmetry preserving robot perception and control through geometry and learning. Frontiers in Robotics and AI 9, 969380

Huang, J., Zeng, W., Xiong, H., Noack, B. R., Hu, G., Liu, S., et al. (2023). Symmetry-informed reinforcement learning and its application to low-level attitude control of quadrotors. IEEE Transactions on Artificial Intelligence 5, 1147–1161

Kasaei, M., Abreu, M., Lau, N., Pereira, A., and Reis, L. P. (2021). A cpg-based agile and versatile locomotion framework using proximal symmetry loss. arXiv preprint arXiv:2103.00928

Kirsch, L., Flennerhag, S., Van Hasselt, H., Friesen, A., Oh, J., and Chen, Y. (2022). Introducing symmetries to black box meta reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence. vol. 36, 7202–7210

Li, Z., Jin, Y., Apraez, D. O., Semini, C., Liu, P., and Chalvatzaki, G. (2025). Morphologically symmetric reinforcement learning for ambidextrous bimanual manipulation. arXiv preprint arXiv:2505.05287

Lin, Y., Huang, J., Zimmer, M., Guan, Y., Rojas, J., and Weng, P. (2020). Invariant Transform Experience Replay: Data Augmentation for Deep Reinforcement Learning. IEEE Robot. Autom. Lett. 5, 6615–6622. ArXiv:1909.10707 [cs]

Liu, L., van de Panne, M., and Yin, K. (2016). Guided learning of control graphs for physics-based characters. ACM Transactions on Graphics 35

Liu, S., Xu, M., Huang, P., Zhang, X., Liu, Y., Oguchi, K., et al. (2023). Continual vision-based reinforcement learning with group symmetries. In Conference on Robot Learning (PMLR), 222–240

Lu, C., Schroecker, Y., Gu, A., Parisotto, E., Foerster, J., Singh, S., et al. (2023). Structured state space models for in-context reinforcement learning. Advances in Neural Information Processing Systems 36, 47016–47031

Luo, J., Xu, C., Wu, J., and Levine, S. (2025). Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. Science Robotics 10, eads5033

[Dataset] Makoviichuk, D. and Makoviychuk, V. (2021). rl-games: A high-performance framework for reinforcement learning. `https://github.com/Denys88/rl_games`

[Dataset] Makoviychuk, V., Wawrzyniak, L., Guo, Y., Lu, M., Storey, K., Macklin, M., et al. (2021). Isaac gym: High performance gpu-based physics simulation for robot learning

Mittal, M., Rudin, N., Klemm, V., Allshire, A., and Hutter, M. (2024). Symmetry considerations for learning task symmetric robot policies. In 2024 IEEE International Conference on Robotics and Automation (ICRA) (IEEE), 7433–7439

Nguyen, H., Kozuno, T., Beltran-Hernandez, C. C., and Hamaya, M. (2024). Symmetry-aware reinforcement learning for robotic assembly under partial observability with a soft wrist. In 2024 IEEE international conference on robotics and automation (ICRA) (IEEE), 9369–9375

Ordonez-Apraez, D., Martin, M., Agudo, A., and Moreno-Noguer, F. (2023). On discrete symmetries of robotics systems: A group-theoretic and data-driven analysis. arXiv preprint arXiv:2302.10433

Schulman, J., Moritz, P., Levine, S., Jordan, M., and Abbeel, P. (2016). High-dimensional continuous control using generalized advantage estimation. In International Conference on Learning Representations

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal Policy Optimization Algorithms ArXiv:1707.06347 [cs]

Singh, S., Russell, R. P., and Wensing, P. M. (2023). Analytical second-order derivatives of rigid-body contact dynamics: Application to multi-shooting ddp. In 2023 22nd IEEE-RAS International Conference on Humanoid Robots (Humanoids) (IEEE), 1–8. doi:10.1109/HUMANOIDS57100.2023.10375214

Sonmez, Y., Junnarkar, N., and Arcak, M. (2024). Exploiting symmetry in dynamics for model-based reinforcement learning with asymmetric rewards. IEEE Control Systems Letters 8, 1180–1185

Su, Z., Huang, X., Ordoñez-Apraez, D., Li, Y., Li, Z., Liao, Q., et al. (2024). Leveraging symmetry in rl-based legged locomotion control. In 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE), 6899–6906. doi:10.1109/iros58592.2024.10802439

Sun, J., Curtis, A., You, Y., Xu, Y., Koehle, M., Chen, Q., et al. (2025). Arch: Hierarchical hybrid learning for long-horizon contact-rich robotic assembly. In Proceedings of The 9th Conference on Robot Learning. vol. 305 of Proceedings of Machine Learning Research, 2628–2642

Sutton, R. and Barto, A. (1998). Reinforcement Learning: An Introduction (Cambridge, MA: MIT Press)

Wang, D., Hart, S., Surovik, D., Kelestemur, T., Huang, H., Zhao, H., et al. (2025). Equivariant diffusion policy. In Conference on Robot Learning (PMLR), 48–69

Wang, D., Walters, R., and Platt, R. (2022). $\mathrm{SO}(2)$-equivariant reinforcement learning. In International Conference on Learning Representations

Yan, S., Zhang, B., Zhang, Y., Boedecker, J., and Burgard, W. (2024). Learning continuous control with geometric regularity from robot intrinsic symmetry. In 2024 IEEE International Conference on Robotics and Automation (ICRA) (IEEE), 49–55

Yu, C., Velu, A., Vinitsky, E., Gao, J., Wang, Y., Bayen, A., et al. (2022). The surprising effectiveness of ppo in cooperative multi-agent games. Advances in neural information processing systems 35, 24611–24624

Yu, W., Turk, G., and Liu, C. K. (2018). Learning symmetric and low-energy locomotion. ACM Transactions on Graphics (TOG) 37, 1–12

Yu, X., Shi, R., Feng, P., Tian, Y., Li, S., Liao, S., et al. (2024). Leveraging partial symmetry for multi-agent reinforcement learning. In Proceedings of the AAAI Conference on Artificial Intelligence. vol. 38, 17583–17590

Zhang, C., Rudin, N., Hoeller, D., and Hutter, M. (2024). Learning agile locomotion on risky terrains. In 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS) (IEEE), 11864–11871

Zinkevich, M. and Balch, T. R. (2001). Symmetry in markov decision processes and its implications for single agent and multiagent learning. In Proceedings of the Eighteenth International Conference on Machine Learning (San Francisco, CA, USA: Morgan Kaufmann Publishers Inc.), ICML '01, 632

Table 23. Joint list of Bi-DexHands

ID	Name	Side	F_{joint}	P_{joint}
0	Left Wrist Joint0	L	1	20
1	Left Wrist Joint1	L	1	21
2	Left First Finger Joint0	L	1	22
3	Left First Finger Joint1	L	1	23
4	Left First Finger Joint2	L	1	24
5	Left Middle Finger Joint0	L	1	25
6	Left Middle Finger Joint1	L	1	26
7	Left Middle Finger Joint2	L	1	27
8	Left Ring Finger Joint0	L	1	28
9	Left Ring Finger Joint1	L	1	29
10	Left Ring Finger Joint2	L	1	30
11	Left Little Finger Joint0	L	1	31
12	Left Little Finger Joint1	L	1	32
13	Left Little Finger Joint2	L	1	33
14	Left Little Finger Joint3	L	1	34
15	Left Thumb Joint0	L	1	35
16	Left Thumb Joint1	L	1	36
17	Left Thumb Joint2	L	1	37
18	Left Thumb Joint3	L	1	38
19	Left Thumb Joint4	L	1	39
20	Right Wrist Joint0	R	1	0
21	Right Wrist Joint1	R	1	1
22	Right First Finger Joint0	R	1	2
23	Right First Finger Joint1	R	1	3
24	Right First Finger Joint2	R	1	4
25	Right Middle Finger Joint0	R	1	5
26	Right Middle Finger Joint1	R	1	6
27	Right Middle Finger Joint2	R	1	7
28	Right Ring Finger Joint0	R	1	8
29	Right Ring Finger Joint1	R	1	9
30	Right Ring Finger Joint2	R	1	10
31	Right Little Finger Joint0	R	1	11
32	Right Little Finger Joint1	R	1	12
33	Right Little Finger Joint2	R	1	13
34	Right Little Finger Joint3	R	1	14
35	Right Thumb Joint0	R	1	15
36	Right Thumb Joint1	R	1	16
37	Right Thumb Joint2	R	1	17
38	Right Thumb Joint3	R	1	18
39	Right Thumb Joint4	R	1	19
ID	Name	Side	F_{base}	P_{base}
40	Left Base Force X	L	1	46
41	Left Base Force Y	L	-1	47
42	Left Base Force Z	L	1	48
43	Left Base Torque X	L	-1	49
44	Left Base Torque Y	L	1	50
45	Left Base Torque Z	L	-1	51
46	Right Base Force X	R	1	40
47	Right Base Force Y	R	-1	41
48	Right Base Force Z	R	1	42
49	Right Base Torque X	R	-1	43
50	Right Base Torque Y	R	1	44
51	Right Base Torque Z	R	-1	45
Total	52			

Table 24. Intrinsic states and symmetry operators of Bi-DexHands

Symbol	Dimension	Description	F_{intr}	P_{intr}
$\mathbf{q}$	48	Joint positions	F_{joint}	P_{joint}
$\dot{\mathbf{q}}$	48	Joint velocities	F_{joint}	P_{joint}
$\boldsymbol{\tau}$	48	Joint torques	F_{joint}	P_{joint}
$\{\mathbf{f}_k^L\}_{k=0}^4$	65	Left-hand fingertip states	*1	$\mathrm{to}(\{\mathbf{f}_k^R\})$
$\{\mathbf{f}_k^R\}_{k=0}^4$	65	Right-hand fingertip states	*1	$\mathrm{to}(\{\mathbf{f}_k^L\})$
$\{\mathbf{s}_k^L\}_{k=0}^4$	30	Left-hand fingertip force	*2	$\mathrm{to}(\{\mathbf{s}_k^R\})$
$\{\mathbf{s}_k^R\}_{k=0}^4$	30	Right-hand fingertip force	*2	$\mathrm{to}(\{\mathbf{s}_k^L\})$
$\mathbf{p}$	12	Hand base poses	F_{base}	P_{base}
$\mathbf{a}_{\mathrm{prev}}$	26	Previous action	$F_{\mathrm{joint}} \circ F_{\mathrm{base}}$	$P_{\mathrm{joint}} \circ P_{\mathrm{base}}$
Total	42			

*1: The fingertip state $\mathbf{f}_k^{\{L|R\}} \in \mathbb{R}^{13}$ contains the position, orientation (represented as a quaternion), linear velocity, and angular velocity of each fingertip. The flipping operator for a single fingertip is $\mathrm{flip}(1, 4, 6, 8, 10, 12)$.

*2: The fingertip force state $\mathbf{s}_k^{\{L|R\}}$ contains the force and torque at each fingertip. The flipping operator for a single fingertip is $\mathrm{flip}(1, 3, 5)$.

Table 25. Extrinsic states and symmetry operators of *Door Open Inward* by Bi-DexHands

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{p}_{\mathrm{LHandle}}$	3	Left door-handle position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{RHandle}})$
$\mathbf{p}_{\mathrm{RHandle}}$	3	Right door-handle position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{LHandle}})$
Total	6			

Table 26. Extrinsic states and symmetry operators of *Lift Underarm* by Bi-DexHands

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{p}_{\mathrm{pot}}$	3	Pot position	$\mathrm{flip}(1)$	-
$\mathbf{R}_{\mathrm{pot}}$	4	Pot orientation quaternion	$\mathrm{flip}(1, 3)$	-
$\mathbf{v}_{\mathrm{pot}}$	3	Pot linear velocity	$\mathrm{flip}(1)$	-
$\boldsymbol{\omega}_{\mathrm{pot}}$	3	Pot angular velocity	$\mathrm{flip}(0, 2)$	-
$\mathbf{p}_{\mathrm{LHandle}}$	3	Left pot-handle position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{RHandle}})$
$\mathbf{p}_{\mathrm{RHandle}}$	3	Right pot-handle position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{LHandle}})$
Total	19			

Table 27. Extrinsic states and symmetry operators of *Two Catch Underarm* by Bi-DexHands

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{p}_{\mathrm{LBall}}$	3	Left ball position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{RBall}})$
$\mathbf{R}_{\mathrm{LBall}}$	4	Left ball orientation quaternion	$\mathrm{flip}(1, 3)$	$\mathrm{to}(\mathbf{R}_{\mathrm{RBall}})$
$\mathbf{v}_{\mathrm{LBall}}$	3	Left ball linear velocity	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{v}_{\mathrm{RBall}})$
$\boldsymbol{\omega}_{\mathrm{LBall}}$	3	Left ball angular velocity	$\mathrm{flip}(0, 2)$	$\mathrm{to}(\boldsymbol{\omega}_{\mathrm{RBall}})$
$\mathbf{p}_{\mathrm{LGoal}}$	3	Left goal position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{RGoal}})$
$\mathbf{R}_{\mathrm{LGoal}}$	4	Left goal orientation quaternion	$\mathrm{flip}(1, 3)$	$\mathrm{to}(\mathbf{R}_{\mathrm{RGoal}})$
$\hat{\mathbf{R}}_{\mathrm{LGoal}}$	4	Left goal orientation quaternion in left ball frame	$\mathrm{flip}(1, 3)$	$\mathrm{to}(\hat{\mathbf{R}}_{\mathrm{RGoal}})$
$\mathbf{p}_{\mathrm{RBall}}$	3	Right ball position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{LBall}})$
$\mathbf{R}_{\mathrm{RBall}}$	4	Right ball orientation quaternion	$\mathrm{flip}(1, 3)$	$\mathrm{to}(\mathbf{R}_{\mathrm{LBall}})$
$\mathbf{v}_{\mathrm{RBall}}$	3	Right ball linear velocity	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{v}_{\mathrm{LBall}})$
$\boldsymbol{\omega}_{\mathrm{RBall}}$	3	Right ball angular velocity	$\mathrm{flip}(0, 2)$	$\mathrm{to}(\boldsymbol{\omega}_{\mathrm{LBall}})$
$\mathbf{p}_{\mathrm{RGoal}}$	3	Right goal position	$\mathrm{flip}(1)$	$\mathrm{to}(\mathbf{p}_{\mathrm{LGoal}})$
$\mathbf{R}_{\mathrm{RGoal}}$	4	Right goal orientation quaternion	$\mathrm{flip}(1, 3)$	$\mathrm{to}(\mathbf{R}_{\mathrm{LGoal}})$
$\hat{\mathbf{R}}_{\mathrm{RGoal}}$	4	Right goal orientation quaternion in right ball frame	$\mathrm{flip}(1, 3)$	$\mathrm{to}(\hat{\mathbf{R}}_{\mathrm{LGoal}})$
Total	26			

Table 28. Hyperparameters in training of Bi-DexHands

Parameter	Value
Actor hidden layers	[512, 256, 128]
Critic hidden layer	[512, 256, 128]
Activation	ELU
Clip range	0.2
Discount factor	0.96
GAE discount factor	0.95
Desired KL divergence	0.016
Learning rate	3×10^{-4}
Number of Environments	2048
Steps per iteration	8
Training iterations	
-*Two Catch Underarm*	12000
- Others	6000

Table 29. Joint list of Quadruped Robot

ID	Name	Side	F_{joint}	P_{joint}
0	FL Hip Joint	L	-1	3
1	FL Thigh Joint	L	1	4
2	FL Calf Joint	L	1	5
3	FR Hip Joint	R	-1	0
4	FR Thigh Joint	R	1	1
5	FR Calf Joint	R	1	2
6	RL Hip Joint	L	-1	9
7	RL Thigh Joint	L	1	10
8	RL Calf Joint	L	1	11
9	RR Hip Joint	R	-1	6
10	RR Thigh Joint	R	1	7
11	RR Calf Joint	R	1	8
Total	12			

Table 30. Intrinsic states and symmetry operators of Quadruped robot

Symbol	Dimension	Description	F_{intr}	P_{intr}
$\mathbf{v}$	3	Base linear velocity	$\text{flip}(1)$	-
$\boldsymbol{\omega}$	3	Base angular velocity	$\text{flip}(0, 2)$	-
$\mathbf{g}$	3	Projected gravity direction	$\text{flip}(1)$	-
$\mathbf{q}$	12	Joint positions	F_{joint}	P_{joint}
$\dot{\mathbf{q}}$	12	Joint velocities	F_{joint}	P_{joint}
$\mathbf{a}_{\text{prev}}$	12	Previous actions	F_{joint}	P_{joint}
Total	45			

Table 31. Extrinsic states and symmetry operators of *Locomotion* by Full-body Humanoid

Symbol	Dimension	Description	F_{extr}	P_{extr}
$\mathbf{v}_{\text{target}}$	2	Target linear velocity	$\text{flip}(1)$	-
$\boldsymbol{\omega}_{\text{target}}$	1	Target angular	$\text{flip}(0)$	-
Total	3			

Table 32. Hyperparameters in training of Quadruped robot

Parameter	Value
Actor hidden layers	[256, 128, 64]
Critic hidden layer	[256, 128, 64]
Activation	ELU
Clip range	0.2
Discount factor	0.99
GAE discount factor	0.95
Desired KL divergence	0.008
Learning rate	3×10^{-4}
Number of Environments	4096
Steps per iteration	24
Training iterations	1000

Table 33. Extrinsic states and symmetry operators of *Block Stack*

Symbol	Dimension	Description	F_{extr}
$\mathbf{p}_{\mathrm{LBlock}}$	3	Left block position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{LBlock}}$	4	Left block orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{LBlock}}$	3	Left block linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{LBlock}}$	3	Left block angular velocity	$\mathrm{flip}(0,2)$
$\mathbf{p}_{\mathrm{RBlock}}$	3	Right block position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{RBlock}}$	4	Right block orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{RBlock}}$	3	Right block linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{RBlock}}$	3	Right block angular velocity	$\mathrm{flip}(0,2)$
Total	26		

Table 34. Extrinsic states and symmetry operators of *Grasp and Place*

Symbol	Dimension	Description	F_{extr}
$\mathbf{p}_{\mathrm{Block}}$	3	Block position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Block}}$	4	Block orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{Block}}$	3	Block linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{Block}}$	3	Block angular velocity	$\mathrm{flip}(0,2)$
$\mathbf{p}_{\mathrm{Cup}}$	3	Cup position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Cup}}$	4	Cup orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{Cup}}$	3	Cup linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{Cup}}$	3	Cup angular velocity	$\mathrm{flip}(0,2)$
Total	26		

Table 35. Extrinsic states and symmetry operators of *Scissors*

Symbol	Dimension	Description	F_{extr}
$\mathbf{p}_{\mathrm{Scissors}}$	3	Scissors position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Scissors}}$	4	Scissors orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{Scissors}}$	3	Scissors linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{Scissors}}$	3	Scissors angular velocity	$\mathrm{flip}(0,2)$
$\mathbf{p}_{\mathrm{LHandle}}$	3	Left handle position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{RHandle}}$	3	Right handle position	$\mathrm{flip}(1)$
Total	16		

Table 36. Extrinsic states and symmetry operators of *Pen*

Symbol	Dimension	Description	F_{extr}
$\mathbf{p}_{\mathrm{Pen}}$	3	Pen position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Pen}}$	4	Pen orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{Pen}}$	3	Pen linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{Pen}}$	3	Pen angular velocity	$\mathrm{flip}(0,2)$
$\mathbf{p}_{\mathrm{Grip}}$	3	Pen grip position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Cap}}$	3	Cap position	$\mathrm{flip}(1)$
Total	16		

Table 37. Extrinsic states and symmetry operators of *Catch Underarm* and *Catch Abreast*

Symbol	Dimension	Description	F_{extr}
$\mathbf{p}_{\mathrm{Ball}}$	3	Ball position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Ball}}$	4	Ball orientation quaternion	$\mathrm{flip}(1,3)$
$\mathbf{v}_{\mathrm{Ball}}$	3	Ball linear velocity	$\mathrm{flip}(1)$
$\boldsymbol{\omega}_{\mathrm{Ball}}$	3	Ball angular velocity	$\mathrm{flip}(0,2)$
$\mathbf{p}_{\mathrm{Goal}}$	3	Goal position	$\mathrm{flip}(1)$
$\mathbf{R}_{\mathrm{Goal}}$	4	Goal orientation quaternion	$\mathrm{flip}(1,3)$
$\hat{\mathbf{R}}_{\mathrm{Goal}}$	4	Goal orientation quaternion in ball frame	$\mathrm{flip}(1,3)$
Total	24		

Table 38. Hyperparameters in training of asymmetric tasks

Parameter	Value
Actor hidden layers	[512, 256, 128]
Critic hidden layer	[512, 256, 128]
Activation	ELU
Clip range	0.2
Discount factor	0.96
GAE discount factor	0.95
Desired KL divergence	0.016
Learning rate	3×10^{-4}
Number of Environments	2048
Steps per iteration	8
Training iterations	6000