

500 A Environmental Details

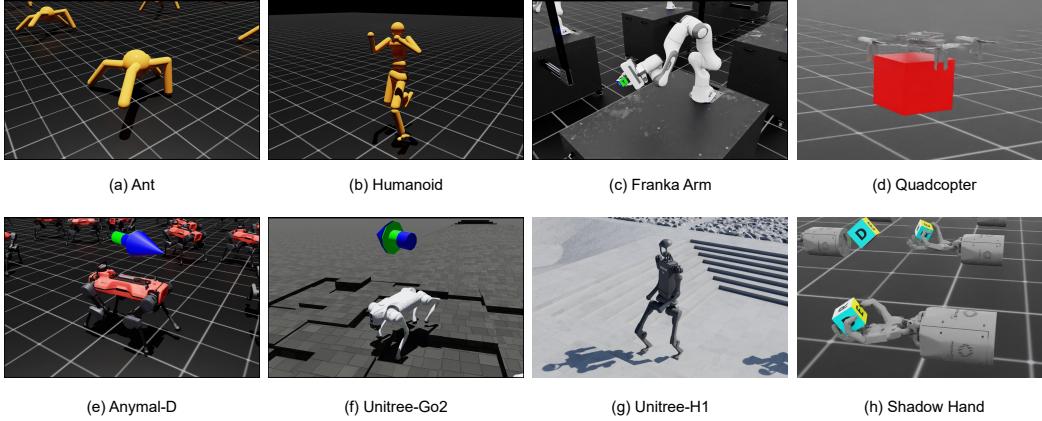


Figure 5: Eight IsaacLab benchmark visualizations, eight images from <https://isaac-sim.github.io/IsaacLab/main/source/overview/environments.html>. From (a) to (h) are Isaac-Ant-v0, Isaac-Humanoid-v0, Isaac-Lift-Cube-Franka-v0, Isaac-Quadcopter-Direct-v0, Isaac-Velocity-Flat-Anymal-D-v0, Isaac-Velocity-Rough-Unitree-Go2-v0, Isaac-Velocity-Rough-H1-v0, and Isaac-Repose-Cube-Shadow-Direct-v0.

IsaacLab¹ is a unified and modular framework for robot learning, designed to streamline common workflows in robotics research, including reinforcement learning, imitation learning, and motion planning. The framework leverages NVIDIA Isaac Sim for high-fidelity simulation and benefits from PhysX’s GPU-accelerated physics engine as well as a tile-based rendering API for vectorized rendering.

As displayed in Figure 5, we selected 8 representative and challenging environments, which can be roughly divided into the following categories according to the official manual of IsaacLab <https://isaac-sim.github.io/IsaacLab/main/source/overview/environments.html>:

1. Classic: These classic tasks are derived from the IsaacGymEnv implementation of MuJoCo-style environments, providing standardized benchmarks for locomotion and control in continuous action spaces.
 - **Isaac-Ant-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{60 \times 8}$. The task involves controlling a MuJoCo Ant robot to move in a specified direction.
 - **Isaac-Humanoid-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{87 \times 21}$. The objective is to control a MuJoCo Humanoid robot to move in a specified direction.
2. Manipulation: These environments involve manipulation tasks performed by a fixed-base robotic arm or a dexterous hand, such as object reaching, grasping, or in-hand rotation.
 - **Isaac-Lift-Cube-Franka-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{36 \times 8}$. The task involves controlling a Franka Emika robot arm to pick up a cube and transport it to a randomly sampled target position.
 - **Isaac-Repose-Cube-Shadow-Direct-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{157 \times 20}$. The task requires using a Shadow Dexterous Hand to perform in-hand reorientation of a cube to match a target orientation.
3. Locomotion: This category includes legged locomotion environments that challenge agents to coordinate multiple degrees of freedom to achieve stable and directed movement. Tasks include forward walking, turning, and terrain adaptation, typically implemented with humanoid robots.
 - **Isaac-Velocity-Flat-Anymal-D-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{48 \times 12}$. The task requires controlling an ANYmal D quadruped robot to follow a commanded velocity trajectory on flat terrain.

¹<https://isaac-sim.github.io/IsaacLab/main/index.html>

531 • **Isaac-Velocity-Rough-Unitree-Go2-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in$
532 $\mathbb{R}^{235 \times 12}$. The task involves controlling a Unitree Go2 quadruped robot to follow a
533 commanded velocity trajectory over rough terrain.

534 • **Isaac-Velocity-Rough-H1-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{256 \times 19}$. The
535 task involves controlling a Unitree H1 humanoid robot to follow a commanded velocity
536 trajectory over rough terrain.

537 4. Others:

538 • **Isaac-Quadcopter-Direct-v0.** The state and action spaces are $(\mathcal{S}, \mathcal{A}) \in \mathbb{R}^{12 \times 4}$. The
539 task is to control quadrotors to fly and hover at a designated goal position by applying
540 thrust commands.

541 Moreover, based on the IsaacLab Engine, the an anonymous link for the visualization of our experi-
542 mental results is shown in `anonymous-project365.github.io`.

543 B Training setups

544 B.1 Hardware Configurations

545 All experiments were carried out on a server equipped with two Intel Xeon Gold 6430 CPUs (32
546 cores per socket, 64 threads total per CPU, 128 threads total), with a base frequency of 2.1 GHz
547 and a maximum turbo frequency of 3.4 GHz. The system supports 52-bit physical and 57-bit virtual
548 addressing. For GPU acceleration, we used 8 NVIDIA GeForce RTX 4090 D GPUs, each with 24
549 GB of GDDR6X memory, connected via PCIe. The GPUs support CUDA 12.8 and were operating
550 under the NVIDIA driver version 570.124.04. The machine is configured with NUMA topology
551 across 2 nodes, each handling 64 logical CPUs. No GPU MIG (Multi-Instance GPU) or ECC was
552 enabled during the experiments.

553 B.2 Reinforcement Learning Framework in IsaacLab

554 IsaacLab provides native integration with several reinforcement learning libraries, in-
555 cluding RSL-RL https://github.com/leggedrobotics/rsl_rl, RL-Games https://github.com/Denys88/rl_games, SKRL <https://skrl.readthedocs.io/en/latest/>, and
556 Stable-Baselines3 <https://stable-baselines3.readthedocs.io/en/master/index.html>,
557 each of which exposes a distinct API for agent-environment interaction. In this
558 work, we adopt SKRL as our primary framework for implementing baseline algo-
559 rithms. SKRL is an open-source, modular, and extensible library that provides stan-
560 dardized implementations of widely used reinforcement learning methods. Specifi-
561 cally, we utilize its implementations of PPO [https://skrl.readthedocs.io/en/latest/](https://skrl.readthedocs.io/en/latest/api/agents/ppo.html)
562 [api/agents/ppo.html](https://skrl.readthedocs.io/en/latest/api/agents/ppo.html), SAC [https://skrl.readthedocs.io/en/latest/api/agents/](https://skrl.readthedocs.io/en/latest/api/agents/sac.html)
563 [sac.html](https://skrl.readthedocs.io/en/latest/api/agents/sac.html), TD3 <https://skrl.readthedocs.io/en/latest/api/agents/td3.html>, and
564 DDPG <https://skrl.readthedocs.io/en/latest/api/agents/ddpg.html> for baseline
565 evaluation. For algorithms such as QVPO and DACER, we design custom environment wrap-
566 pers to adapt the existing interface without modifying the underlying simulation environments. Our
567 proposed method, GenPO, is implemented using RSL-RL, a lightweight and high-performance
568 framework tailored for robotics and continuous control, which emphasizes computational efficiency
569 and ease of deployment.

571 B.3 Hyperparameters

572 Tables 2 and Table 10 summarize the hyperparameter configurations used across our experiments. For
573 the baseline algorithms: PPO, SAC, TD3, and DDPG, we adopt the default hyperparameter settings
574 provided by the SKRL library. For our proposed method, GenPO, we align its hyperparameter
575 configuration with that of PPO to ensure a fair and controlled comparison.

Table 2: Hyper-parameters used in the IsaacLab-Ant-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[400,200,100]	[400,200,100]	[256,256,256]	[256,256,256]	[400,200,100]	[400,200,100]	[400,200,100]
hidden layers in critic network	[400,200,100]	[400,200,100]	[256,256,256]	[256,256,256]	[400,200,100]	[400,200,100]	[400,200,100]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	1×10^{-3}	1×10^{-3}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Learning rate for critic	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.01	0.01	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 3: Hyper-parameters used in the IsaacLab-Humanoid-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[400,200,100]	[400,200,100]	[256,256,256]	[256,256,256]	[400,200,100]	[400,200,100]	[400,200,100]
hidden layers in critic network	[400,200,100]	[400,200,100]	[256,256,256]	[256,256,256]	[400,200,100]	[400,200,100]	[400,200,100]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	5×10^{-4}	5×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Learning rate for critic	5×10^{-4}	5×10^{-4}	3×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.01	0.01	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 4: Hyper-parameters used in the Isaac-Lift-Cube-Franka-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[256,128,64]	[256,128,64]	[256,256,256]	[256,256,256]	[256, 128, 64]	[256, 128, 64]	[256, 128, 64]
hidden layers in critic network	[256,128,64]	[256,128,64]	[256,256,256]	[256,256,256]	[256, 128, 64]	[256, 128, 64]	[256, 128, 64]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.98	0.98	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	1×10^{-4}	1×10^{-4}	3×10^{-4}	5×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Learning rate for critic	1×10^{-4}	1×10^{-4}	3×10^{-4}	3×10^{-4}	1×10^{-4}	1×10^{-4}	1×10^{-4}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.006	0.006	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 5: Hyper-parameters used in the Isaac-Quadcopter-Direct-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[64,64]	[64,64]	[256,256,256]	[256,256,256]	[64, 64]	[64, 64]	[64, 64]
hidden layers in critic network	[64,64]	[64,64]	[256,256,256]	[256,256,256]	[64, 64]	[64, 64]	[64, 64]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	5×10^{-4}	5×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Learning rate for critic	5×10^{-4}	5×10^{-4}	3×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.01	0.01	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 6: Hyper-parameters used in the Isaac-Velocity-Flat-Anymal-D-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[128,128,128]	[128,128,128]	[256,256,256]	[256,256,256]	[128, 128, 128]	[128, 128, 128]	[128, 128, 128]
hidden layers in critic network	[128,128,128]	[128,128,128]	[256,256,256]	[256,256,256]	[128, 128, 128]	[128, 128, 128]	[128, 128, 128]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	1×10^{-3}	1×10^{-3}	3×10^{-4}	5×10^{-4}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Learning rate for critic	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.005	0.005	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 7: Hyper-parameters used in the Isaac-Velocity-Rough-Unitree-Go2-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[512, 256, 128]	[512, 256, 128]	[256,256,256]	[256,256,256]	[512, 256, 128]	[512, 256, 128]	[512, 256, 128]
hidden layers in critic network	[512, 256, 128]	[512, 256, 128]	[256,256,256]	[256,256,256]	[512, 256, 128]	[512, 256, 128]	[512, 256, 128]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	1×10^{-3}	1×10^{-3}	3×10^{-4}	5×10^{-4}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Learning rate for critic	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.01	0.01	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 8: Hyper-parameters used in the Isaac-Velocity-Rough-H1-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[512, 256, 128]	[512, 256, 128]	[256,256,256]	[256,256,256]	[512, 256, 128]	[512, 256, 128]	[512, 256, 128]
hidden layers in critic network	[512, 256, 128]	[512, 256, 128]	[256,256,256]	[256,256,256]	[512, 256, 128]	[512, 256, 128]	[512, 256, 128]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	1×10^{-3}	1×10^{-3}	3×10^{-4}	5×10^{-4}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Learning rate for critic	1×10^{-3}	1×10^{-3}	3×10^{-4}	3×10^{-4}	1×10^{-3}	1×10^{-3}	1×10^{-3}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.01	0.01	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.01	0.01	N/A	N/A	N/A	N/A	N/A

Table 9: Hyper-parameters used in the Isaac-Repose-Cube-Shadow-Direct-v0.

Hyperparameters	GenPO	PPO	QVPO	DACER	SAC	TD3	DDPG
hidden layers in actor network	[512, 512, 256, 128]	[512, 512, 256, 128]	[256,256,256]	[256,256,256]	[512, 512, 256, 128]	[512, 512, 256, 128]	[512, 512, 256, 128]
hidden layers in critic network	[512, 512, 256, 128]	[512, 512, 256, 128]	[256,256,256]	[256,256,256]	[512, 512, 256, 128]	[512, 512, 256, 128]	[512, 512, 256, 128]
Activation	mish	elu	mish	relu	mish	mish	mish
Batch size	4096	4096	4096	4096	4096	4096	4096
Use GAE	True	True	N/A	N/A	N/A	N/A	N/A
Discount for reward γ	0.99	0.99	0.99	0.99	0.99	0.99	0.99
GAE Smoothing Parameter λ	0.95	0.95	N/A	N/A	N/A	N/A	N/A
Learning rate for actor	5×10^{-4}	5×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Learning rate for critic	5×10^{-4}	5×10^{-4}	3×10^{-4}	3×10^{-4}	5×10^{-4}	5×10^{-4}	5×10^{-4}
Actor Critic grad norm	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Memory size	N/A	N/A	1×10^6	1×10^6	1×10^6	1×10^6	1×10^6
Entropy coefficient	0.005	0.005	N/A	N/A	N/A	N/A	N/A
Value loss coefficient	1.0	1.0	1.0	1.0	1.0	1.0	1.0
Noise clip	N/A	N/A	N/A	0.5	N/A	N/A	N/A
Surrogate clip	0.2	0.2	N/A	N/A	N/A	N/A	N/A
Diffusion steps	5	N/A	20	20	N/A	N/A	N/A
Desired KL	0.016	0.016	N/A	N/A	N/A	N/A	N/A

Table 10: Hyper-parameters used in GenPO.

Hyperparameter	Ant	Humanoid	Franka Arm	Quadcopter	Anymal-D	Unitree-Go2	Unitree-H1	Shadow Hand
Compress coefficient λ	0.01	0.01	0.01	0.01	0.01	0.01	0.01	0.01
Time embedding dimension	32	32	32	16	32	32	32	32
Hidden layers in time embedding	[256,256]	[256,256]	[256,256]	[128,128]	[256,256]	[256,256]	[256,256]	[256,256]
Mixing coefficient p	0.9	0.9	0.9	0.9	0.9	0.9	0.9	0.9

C Details of Model Architecture

C.1 Sinusoidal Positional Embedding

During training, instead of directly concatenating the time step t , state s_t , and action a_t as raw input features, we incorporate sinusoidal positional embeddings to encode the temporal component. This approach, inspired by the positional encoding technique commonly used in denoising diffusion probabilistic models (DDPMs), maps each timestep $t \in \{0, 1, \dots, T\}$ to a fixed-dimensional vector $e_t \in \mathbb{R}^d$ as follows:

$$e_t^{(2i)} = \sin\left(\frac{t}{10000^{2i/d}}\right), \quad e_t^{(2i+1)} = \cos\left(\frac{t}{10000^{2i/d}}\right), \quad \text{for } i = 0, 1, \dots, \frac{d}{2} - 1. \quad (13)$$

The resulting embedding e_t is then passed through a multi-layer perceptron (MLP) to allow for non-linear transformation and projection into the model’s feature space. The MLP output is subsequently concatenated with the state s_t and action a_t to form the final input representation:

$$\tilde{x}_t = \text{concat}(s_t, a_t, \text{MLP}(e_t)), \quad (14)$$

which enriches the model input with a smooth, continuous representation of temporal progression. Unlike raw scalar timestep concatenation, sinusoidal embeddings provide a structured encoding that helps the model infer temporal relationships and ordering, even in the absence of explicit recurrence or attention mechanisms.

590 C.2 Actor and Critic Model Architecture

591 In the Isaac-Ant-v0 environment, the actor network, corresponding to the flow policy, is implemented
 592 as a multi-layer perceptron (MLP) with hidden layer dimensions [400,200,100]. The activation
 593 function used is Mish. The input to the actor consists of the concatenation of the current state, action,
 594 and sinusoidal time embedding vectors, resulting in an input dimension of $|\mathcal{A}| + |\mathcal{S}| + |\mathcal{T}|$. The output
 595 dimension is equal to the action dimension $|\mathcal{A}|$, representing the flow-based action refinement. The
 596 critic network is also implemented as an MLP with the same hidden layer structure [400,200,100] and
 597 Mish activation. It receives only the current state as input and outputs a scalar value corresponding to
 598 the estimated state value.

599 D Additional Experiments

600 D.1 Effect of Different Parallel Environments Size

601 To assess the effect of parallelization on the learning efficiency of GenPO, we conducted an experiment
 602 varying the number of parallel environments. We trained GenPO agents using 512, 1024, 2048, 4096,
 603 and 8192 parallel environments, respectively. All other hyperparameters were kept consistent across
 604 these runs.

605 As illustrated in Figure 6, increasing the number of parallel environments generally leads to improved
 606 learning performance. Agents trained with a larger number of environments exhibit faster convergence
 607 and achieve higher final rewards compared to those trained with fewer environments. However, the
 608 performance gain from using 8192 environments over 4096 is marginal. To balance performance and
 609 computational efficiency, we fix the number of parallel environments to 4096 in all experiments.

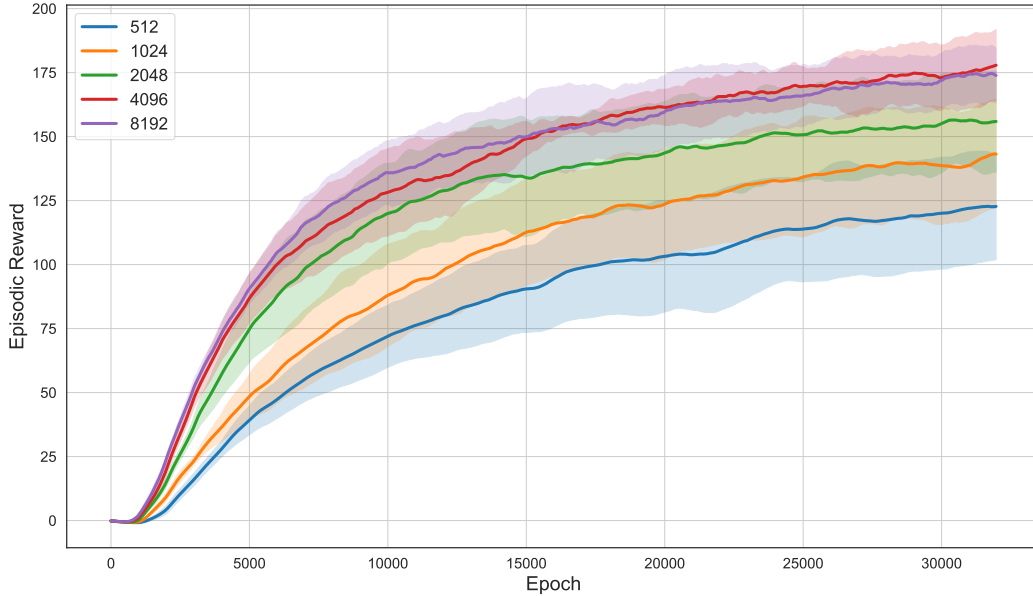


Figure 6: Learning curves for the Isaac-Ant-v0 benchmark with different numbers of parallel environments. Results are averaged over 5 runs. The x-axis denotes training epochs, and the y-axis shows average episodic return with one standard deviation shaded.

610 D.2 Effect of Different Flow Policy Steps

611 We investigate the impact of the number of flow steps, denoted by T , on learning performance. To
 612 this end, we train agents using different values of $T \in \{1, 2, 5, 10, 20\}$, while keeping all other
 613 hyperparameters fixed. As shown in Figure 7, the number of flow steps influences convergence

614 speed and performance stability. Notably, as long as T is not too small (e.g., $T \geq 2$), the overall performance remains robust.

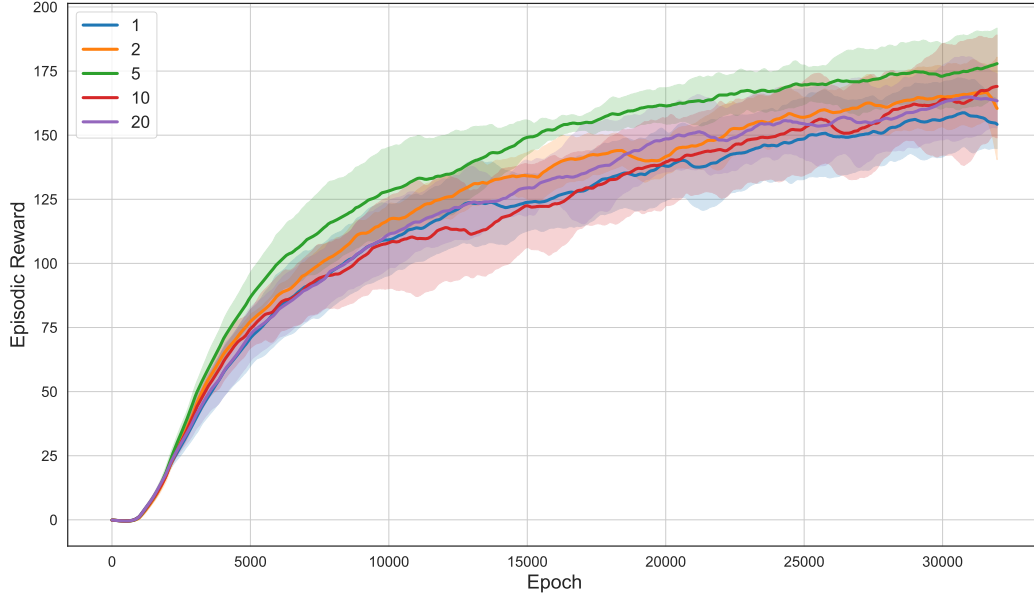


Figure 7: Learning curves for different flow policy time steps on the IsaacLab-Ant-v0 benchmark. Results are averaged over 5 runs. The x-axis denotes training epochs, and the y-axis shows average episodic return with one standard deviation shaded.

615

616 D.3 Effect of Different Operations on Dummy Action Space

617 We study the impact of recovering a single action from the dummy action $\tilde{a} = (x, y)$ by interpolating
618 between the two partial components x, y . Specifically, we construct the real action as $\alpha x + (1 - \alpha) y$
619 and $\alpha \in \{0.0, 0.3, 0.5, 0.7, 1.0\}$ respectively. As illustrated in Figure 8, the best performance
620 is achieved when $\alpha = 0.5$. This observation highlights the benefit of equally leveraging both
621 components of the dummy action. When $\alpha = 0.0$ or $\alpha = 1.0$, only one side of the dummy action
622 space is utilized, leading to limited exploration and suboptimal performance. In contrast, interpolating
623 between both components enhances exploration diversity and improves sample efficiency, which is
624 crucial for effective policy optimization in high-dimensional or multimodal action spaces.

625 D.4 Effect of Time Steps Embedding

626 We investigate the effect of sinusoidal time embeddings on the performance of the flow policy. As
627 shown in Figure 9, incorporating sinusoidal embeddings significantly improves the policy’s ability
628 to utilize temporal information across different stages of the diffusion process. The structured,
629 high-dimensional representation provided by the sinusoidal encoding facilitates better discrimination
630 of time steps, which in turn enhances the model’s capacity to learn effective denoising functions. In
631 contrast, using simple concatenation of the scalar timestep with the state and action fails to provide
632 sufficient temporal structure, resulting in degraded performance. Based on these observations, we
633 adopt sinusoidal positional embeddings to encode time steps in all subsequent experiments.

634 D.5 Training and Inference Time

635 To validate the computational efficiency, we also conduct comparison experiments on the proposed
636 GenPO with 6 other baselines, which respectively evaluate the inference, training and parallel training
637 time on single/multiple GPUs, as shown in Figure 10, Figure 11 and Figure 12. It can be observed
638 that, while the GenPO’s inference time is slightly longer than that of Gaussian policies due to multiple

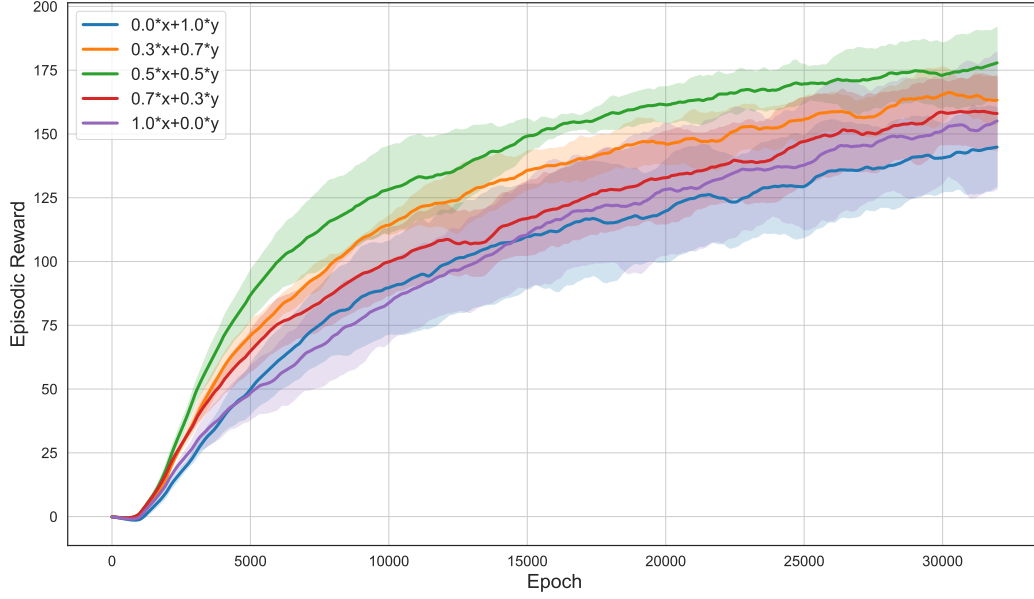


Figure 8: Learning curves for different operations on dummy action space on the IsaacLab-Ant-v0 benchmark. Results are averaged over 5 runs. The x-axis denotes training epochs, and the y-axis shows average episodic return with one standard deviation shaded.

denoising iterations of diffusion, it remains significantly more efficient than previous diffusion-based RL algorithms such as QVPO and DACER. Besides, to our knowledge, the inference time of 2.577ms is more than sufficient for the real-time decision in robot control. Furthermore, in terms of training time, although the performance of GenPO on a single GPU is less than ideal (yet still superior to previous diffusion-based RL methods), it can effectively leverage multi-GPU parallelism to significantly accelerate training. In contrast, the traditional RL algorithm PPO cannot obviously benefit from that.

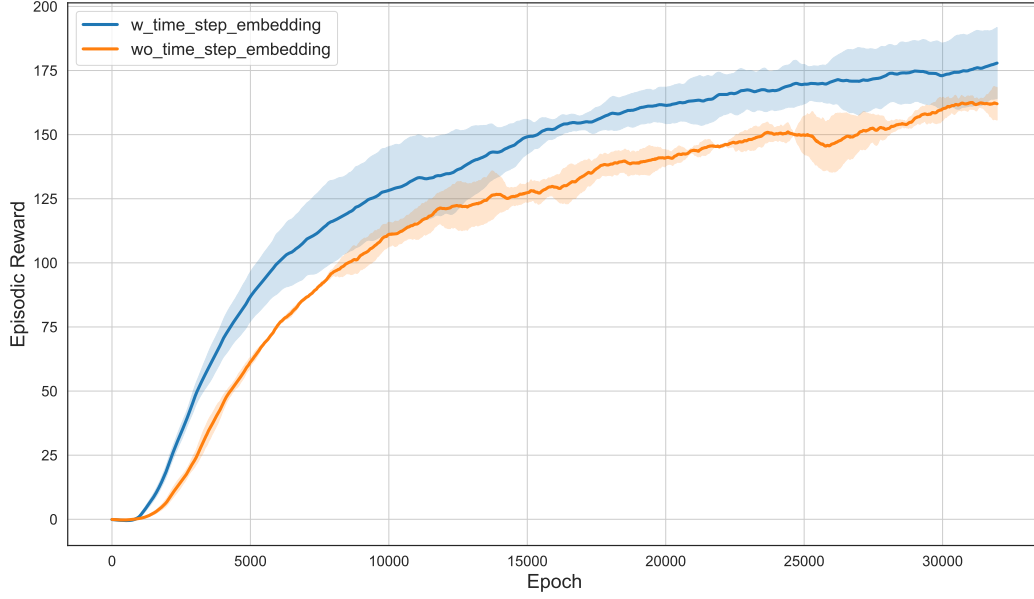


Figure 9: Learning curves on the IsaacLab-Ant-v0 benchmark with or without time step sinusoidal positional embedding. Results are averaged over 5 runs. The x-axis denotes training epochs, and the y-axis shows average episodic return with one standard deviation shaded.

646 E Proof for Lemma 1

647 **Lemma 1. (Change of Variables [3])** Let $f : \mathbb{R}^n \rightarrow \mathbb{R}^n$ is an invertible and smooth mapping. If we
 648 have the random variable $X \sim q(x)$ and the random variable $Y = f(X)$ transformed by function f ,
 649 the distribution $p(y)$ of Y is

$$p(y) = q(x) \left| \det \frac{\partial f}{\partial x} \right|^{-1}. \quad (15)$$

650 *Proof.* Here, we take the one-dimensional random variables X, Y as an example. For a general
 651 n-dimensional case, we can obtain the same result by applying multivariable calculus.

652 Firstly, we can divide f into two categories: (1) f is strictly increasing, (2) f is strictly decreasing,
 653 since f is an invertible and smooth mapping.

654 **Strictly increasing.** For strictly increasing f , the CDF of Y is

$$P(y) = P(Y \leq y) = P(f(X) \leq y) = P(X \leq f^{-1}(y)) = Q(f^{-1}(y)) = Q(x).$$

655 Then, according to $p(x) = \lim_{\epsilon \rightarrow 0} \frac{P(x+\epsilon) - P(x)}{\epsilon}$ and the chain rule, the PDF of the random variable Y is

$$p(y) = q(x) \frac{dx}{dy} = q(x) \left(\det \frac{\partial f}{\partial x} \right)^{-1}.$$

656 **Strictly decreasing.** The proof for f strictly decreasing is analogous. In that case, the PDF of Y is
 657 $p(y) = -q(x) \frac{dx}{dy}$ since $\frac{dx}{dy} < 0$. Hence, we need to add an absolute sign for the final formula, i.e.,

$$658 \quad p(y) = q(x) \left| \det \frac{\partial f}{\partial x} \right|^{-1}. \quad \square$$

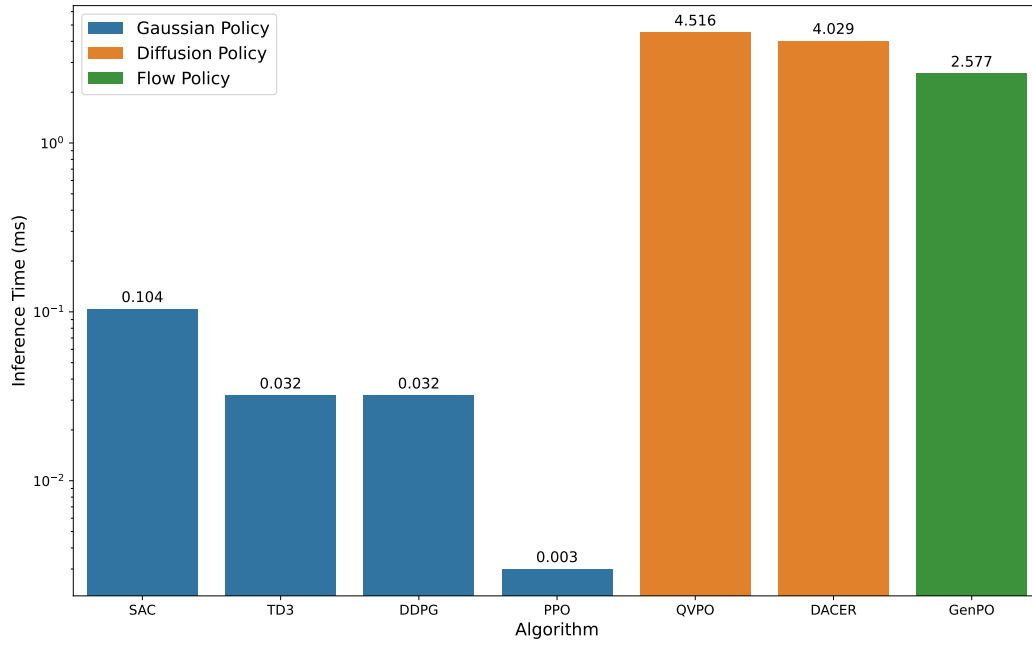


Figure 10: Inference time comparison on Issac-Ant-v0 environment.

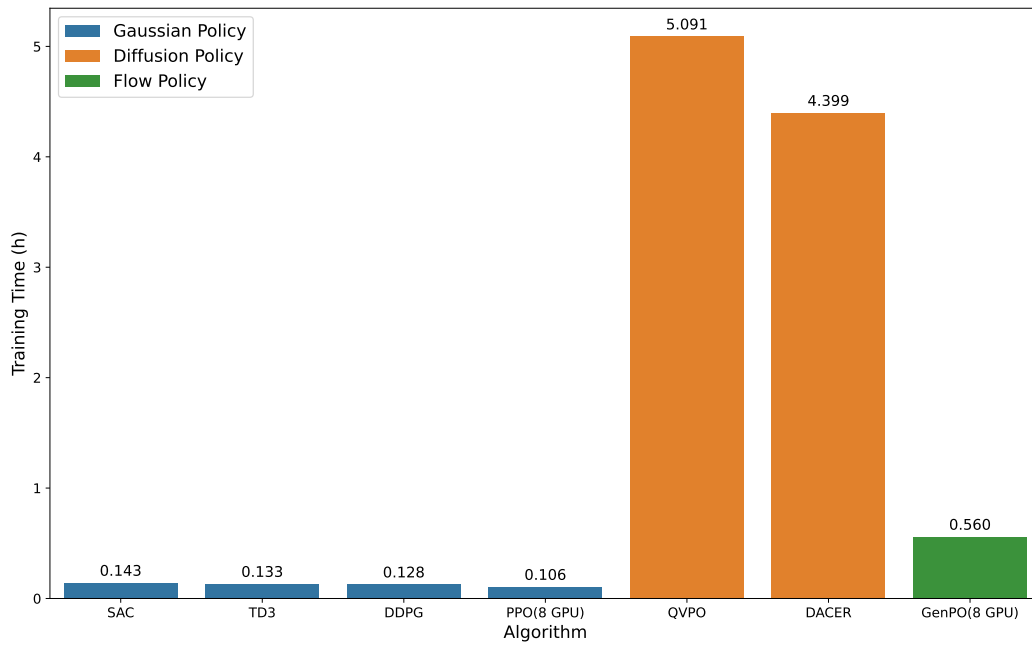


Figure 11: Training time comparison on Issac-Ant-v0 environment.

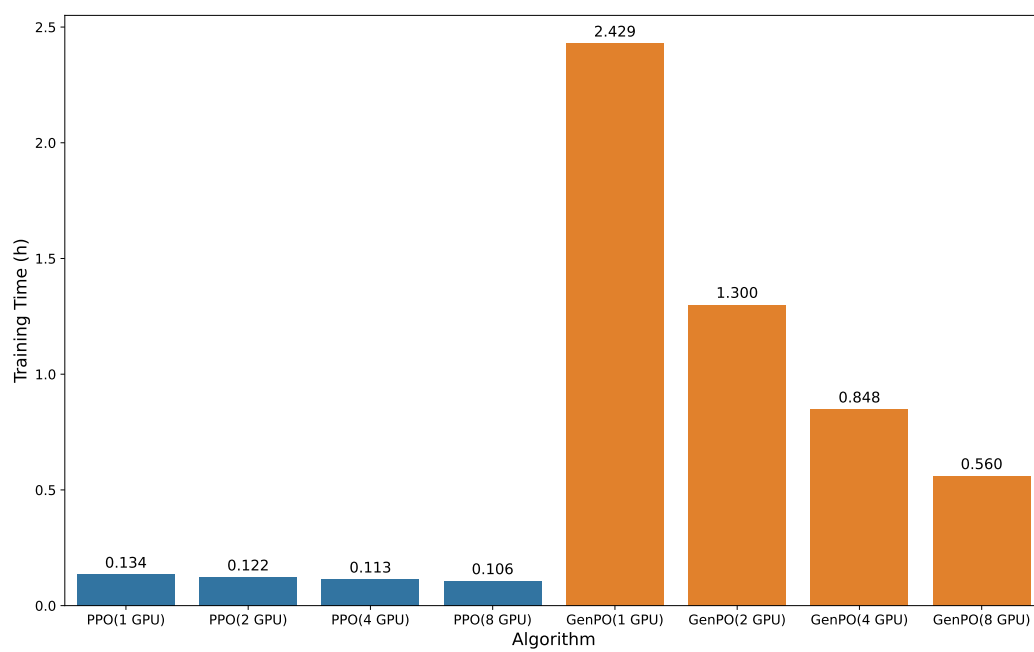


Figure 12: Comparison of training time between PPO and GenPO on the Isaac-And-v0 environment for scaling training on multiple GPUs.