



Research Article

Efficient co-adaptation of humanoid robot design and locomotion control using surrogate-guided optimization

Yidong Du, Xuechao Chen^{*}, Zhangguo Yu, Fei Meng, Zishun Zhou, Yuanxi Zhang, Qingqing Li, Qiang Huang

School of Mechatronic Engineering, Beijing Institute of Technology, Beijing 100081, China

ARTICLE INFO

Article history:

Received 12 April 2025

Revised 30 June 2025

Accepted 13 July 2025

Available online 13 August 2025

Keywords:

Humanoid robot

Design co-adaptation

Reinforcement learning

ABSTRACT

Recent advancements in reinforcement learning (RL) and computational resources have demonstrated the efficacy of data-driven methodologies for robotic locomotion control and physical design optimization, providing a scalable alternative to traditional human-crafted design paradigms. However, existing co-design approaches face a critical challenge: the computational intractability of exploring high-dimensional design spaces, exacerbated by the resource-intensive nature of policy training and candidate design evaluations. To address this limitation, we propose an efficient co-adaptation framework for humanoid robot kinematics optimization. Building on a bi-level optimization architecture that jointly optimizes mechanical designs and control policies, our method achieves computational efficiency through two synergistic strategies: (1) a universal policy generalizable across design variations, and (2) a surrogate-assisted fitness evaluation mechanism. We implement the method with humanoid robot Kuafu, and by experimental results we demonstrate the proposed method effectively reduces the cost and the optimized design can achieve near-optimal performance.

© 2025 The Author(s). Published by Elsevier B.V. on behalf of Shandong University. This is an open access article under the CC BY-NC-ND license (<http://creativecommons.org/licenses/by-nc-nd/4.0/>).

1. Introduction

In recent years, humanoid robots have garnered significant research interest due to their morphologic design and potential for seamless integration into human-centric environments. This biomimetic approach aims to replicate the efficiency of biological locomotion systems, with robust mobility being a critical requirement for real-world deployment. While existing studies have developed stable motion control systems through model-based optimization [1] and online adaptation strategies [2], the emergence of deep learning has further revolutionized locomotion control paradigms for legged robots [3,4].

Beyond motion control, locomotion performance is intrinsically coupled with mechanical design optimization. Biological systems achieve this through co-evolution of form and function, but conventional robotic design processes rely on expert craftsmanship to resolve trade-offs such as inertial distribution, actuator torque-speed characteristics [5], and high-dimensional parameter exploration. These manual approaches face three key limitations: (1) prohibitive iteration costs due to physical prototyping constraints, (2) task-specific optimality requirements, (3) inability to support rapid design adaptation. These challenges highlight the need for efficiently automated robot design methods.

To address the problem of efficient robot design co-adaptation, recent studies focused on applying computational optimization methods to automate the robot behavior and design co-adaptation. Most current studies approach this issue by formulating this problem as a bi-level optimization problem [6–11]. In the inner optimization, the optimal motion controller is determined for the current design instance. Then in the outer optimization, the fitness of the design instance, equipped with the controller, is evaluated and used to optimize the design parameters. For the inner optimization, many works employ Model-free Deep Reinforcement Learning (DRL) method as motion controller [6,7]. And for the outer optimization, gradient-free optimization methods such as Evolutionary Algorithms (EA) are usually used for this problem is non-differentiable in most cases [9,12,13]. However, this paradigm remains computationally limited by high-dimensional design spaces, costly policy training, and expensive fitness evaluations.

Drawing inspiration from the evolutionary co-adaptation of morphology and neuromuscular control within species, where selective pressures drive simultaneous optimization of physical traits and movement strategies, we propose a novel DRL-based co-adaptation framework. First, we adopt the idea of training a universal policy to control robots of varying designs [14,15]. Different to the specialized policy which trains and evaluates on single design instance, universal policy allows the simultaneous training of multiple design instances and therefore the efficient exploration of design space. Second, to make full use of the large

^{*} Corresponding author.

E-mail address: chenxuechao@bit.edu.cn (X. Chen).

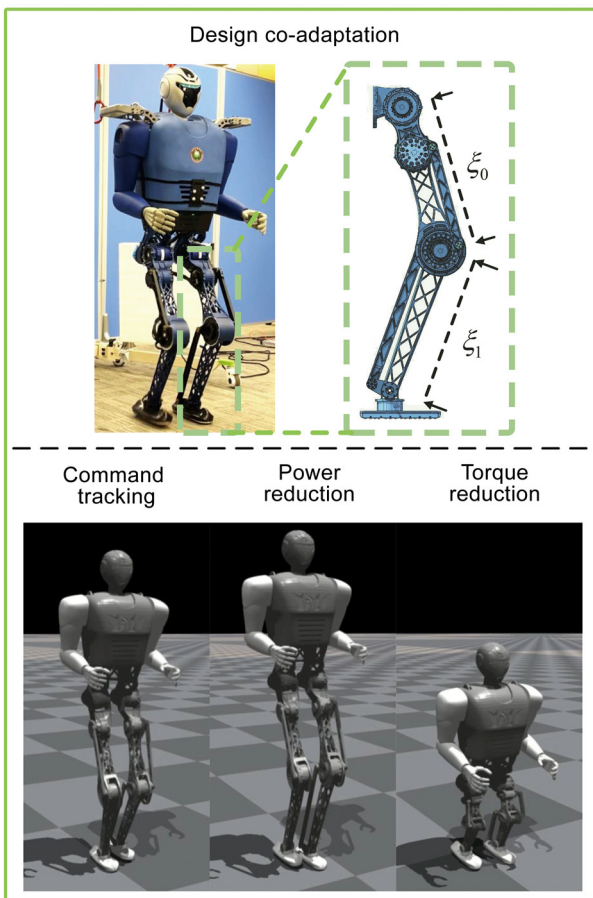


Fig. 1. Examples of optimized designs with respect to different objectives on flat terrains using the proposed method.

amount of interactions of robots with different morphologies during training, we concurrently train specialized pseudo-critics as Surrogate Models [16,17], which is used as fitness estimator during fitness evaluation. The employment of surrogate model can reduce the cost of design instance fitness evaluation. Consequently, the proposed framework significantly lowers the computational expenses and enables efficient humanoid robot design co-adaptation. The examples of optimized results using proposed method is shown in Fig. 1.

The main contributions of this paper are as follows:

- We present a novel DRL-based robot design co-adaptation framework, in which we propose to introduce universal policy controller and critic-based Surrogate Model to enable efficient co-adaptation.
- The proposed algorithm is evaluated by extensive experiments.

The remainder of this paper is organized as follows. Section 2 reviews related works. Section 3 presents our methodology. Section 4 presents the experimental results. Finally, Section 5 concludes this paper.

2. Related works

2.1. RL for legged robot

Researchers recently have successfully applied RL-based policy to both simulated and real legged robot locomotion task

including quadrupedal and bipedal robot platform. The ability of learned end-to-end policy to be deployed on quadrupedal ANYmal robot in complex environments without or with the aid of exteroceptive perception has been demonstrated in [18–20]. Model-based trajectory optimization can also be combined with model-free RL method to achieve robust ANYmal locomotion on sparse terrains [21].

Works have also been presented to show the potential of RL on bipedal and humanoid robots. Siekmann et al. [22] demonstrate the performance of using Recurrent Neural Network (RNN) as locomotion policy on Cassie and claims RNN increases the robustness compared to Multi-layer Perceptron (MLP). In [23], Siekmann et al. achieve a controlled gait pattern using reward function with a periodic signal on Cassie. Li et al. [24] proposes a general dual-history control framework for diverse skills such as walking, running and jumping in an end-to-end manner, and showed its task generalization and compliance to disturbances. Haarnoja et al. [25] use low-cost humanoid robot OP3 to learn movement skills including walking, turning, kicking and finally is capable of playing one-versus-one soccer game. Recently Radosavovic et al. [3] employ the transformer-based controller to achieve humanoid robot Digit locomotion, which represents the first successful attempt to control full-sized humanoid robot using learning-based method. Tang et al. [26] combine imitation learning technique and human locomotion demos to learn natural gaits on humanoid robot. Fu et al. [4] leverage human data to learn humanoid robot motion and autonomous skills such as boxing and folding clothes.

2.2. Robot design co-adaptation

Existing works on robot co-design broadly fall into two categories according to the type of robot motion controller: model-based and model-free methods. Model-based methods normally optimize the robot design toward the best performance under a predefined and model-based controller.

Digumarti et al. [12] apply Covariance Matrix Adaptation Evolution Strategy (CMA-ES) to jointly optimize design parameters and the model-based controller parameters of quadrupedal robot with the objective of maximizing linear speed. Ha et al. [13] also use CMA-ES to jointly optimize the trajectories and design of a planar robot. Genetic Algorithm (GA) is also used in walking robot design optimization, Chadwick et al. [9] present a framework to optimize the leg design for walking robots using GA. Fadini et al. [27] aim to reduce the energy consumption for monoped robot jumping task by finding best trade-offs among structure, actuator and control strategy. Sun et al. [28] propose to use topology optimization method to design fully compliant legs for quadrupedal robots. Other studies focused on manipulation task based robot design [11,29]. Zhu et al. [11] propose a optimization and evaluation framework for 7-dof humanoid robot arm for humanoid manipulation task. Xu et al. [29] present a differentiable rigid body simulation to enable end-to-end differentiable optimization for contact-aware robot design.

However, model-based methods have inherent limitations. The models on which these methods rely are typically simplified, potentially resulting in mismatch with real-world robot. Additionally, the motions or gaits are also often based on predefined patterns. Both of the characteristics make the model-based methods highly handcraft-dependent which makes it difficult to claim the optimized design is global optimum.

Recently, as Reinforcement Learning is gaining increasing research interests for robot control, model-free RL methods also have been shown to be an alternative in robot design optimization, for it does not require predefined model and ability to explore optimal behavior [6,7,10,30,31]. Alvaro et al. Luck

et al. [31] propose to use a dual-network mechanism, which includes an individual network and population network with Actor-Critic architecture, to optimize the simulated robot agents design. The two networks interact with the environment with specific design and integrate the data of all designs to serve as a surrogate to guide the optimization respectively. The mechanism can effectively reduce the cost in robot design co-optimization. In [7], the parameters of Parallel-elastic Actuator (PEA) on ANYmal robot are optimized to find the designs with best energy efficiency on different terrains. Hu et al. [10] present a method using multi-fidelity Bayesian Optimization which is capable of utilizing evaluation of low-fidelity design candidate to explore the design space efficiently to optimize the morphology of OpenAI Gym standardized agents Cheetah and Ant. Different from many existing works that apply bi-level optimization, Ha et al. [30] directly use a variant of policy gradient to learn both the control policy and parameterized design parameters of Ant and Walker. Similarly, Schaff et al. [32] propose to maintain a design distribution which is optimized coupled with control policy. The work most related to ours is [6]. It proposes a design optimization framework which use Meta-RL policy and Genetic Algorithm to quickly adapt to varying designs by optimizing the leg length of ANYmal robot on different terrains. However, the meta-policy still falls into the category of specialized policy, and it also fails to consider the costly design instance evaluation process.

Though this paper does not focus on changing the topological structure of humanoid robot, there are also works involving the optimization of varying-structure virtual agent [8,33–35]. Zhao et al. [33] combine Graph Heuristic Search and Model Predictive Control (MPC) to explore design space formed by a set of fixed body attributes for modular robot. Luck et al. [8] combine Graph Neural Network (GNN) as global performance evaluator and MLP as specific design controller, to optimize legged robot with varying DoFs. Gupta et al. [34] propose DERL which leverages RL to enable robot to evolve to nontrivial robot morphology. Wang et al. [35] present an optimization framework which is a combination of universal policy and evolutionary algorithm. The introduction of universal policy that can control robots of different morphologies significantly shortened the training time.

3. Method

The proposed method aims to achieve efficient co-adaptation of robot motion controllers and morphologies across diverse task environments and objectives. Central to this framework is a bi-level optimization architecture, where the inner loop optimizes the motion controller via reinforcement learning (RL), while the outer loop optimizes the robot design parameters.

For the inner optimization, we leverage advancements in RL to train a universal control policy capable of generalizing across diverse morphologies. This approach enables simultaneous exploration of the design space, significantly improving efficiency compared to traditional model-based controllers that rely on handcrafted adjustments and predefined motion patterns.

The outer optimization addresses the non-differentiable nature of the design-controller interaction by employing gradient-free methods. While Bayesian Optimization (BO) and similar techniques are viable alternatives, we adopt Particle Swarm Optimization (PSO) due to its inherent compatibility with parallelized, batch evaluations of design candidates. This allows efficient exploration of high-dimensional design spaces through swarm evaluation, where particles iteratively converge toward regions of high fitness.

In summary, the two phases of the framework operate iteratively:

Phase 1 Universal Policy Training: A universal policy and pseudo-critic networks, which are used as surrogate models, are

trained concurrently using diverse, randomly sampled design instances.

Phase 2 Design Optimization: The PSO updates the design optimization, based on the evaluation of the fitness of design candidates. The evaluation uses a hybrid approach, which combines computationally expensive real evaluations and low-cost surrogate predictions.

This iterative process progressively narrows the design space toward high-performance regions while maintaining computational tractability. Crucially, the batch evaluation capability of both RL and PSO ensures seamless coordination between the two optimization levels, enabling large-scale exploration of controller and design parameters without prohibitive resource demands. The overview of our method is shown in Fig. 2.

In this section, we present our method of humanoid robot design co-adaptation framework by first reviewing the concept of bi-level optimization and then the details of each phase.

3.1. Preliminary and bi-level design optimization

The application of the bi-level optimization aims to seek optimal design-control pairs (ξ^*, π^*) by searching the design space \mathcal{E} and control space Π . In the inner optimization, or phase 1, we use RL to optimize the controller π with respect to current design ξ . The RL problem can be modeled as Markov Decision Process (MDP). An MDP is defined by a 5-tuple $M = \{S, A, P, r, \gamma\}$, where S is the state space, A is the action space, $P(s_{t+1}|s_t, a_t)$ is the transition model, $r(s_t, a_t, \xi)$ is the reward function, and γ is the discount factor. In practice, the state of MDP can only be partially observed or estimated, so the MDP is modified to be Partially Observable Markov Decision Process (POMDP), with an observation space O and observation function $Z(o_t|s_t)$. The objective of the inner optimization, is to find $\pi_{\theta, \xi}^*$ to maximize the cumulative discounted reward, formulated by

$$\pi_{\theta, \xi}^* = \arg \max_{\pi_{\theta, \xi}} \mathbb{E} \left[\sum_t [\gamma^t r(s_t, a_t, \xi)] \right] \quad (1)$$

With the controller $\pi_{\theta, \xi}^*$, the algorithm can proceed to the outer optimization of the design ξ . The goal of design optimization is to maximize the fitness functions $f(\xi, \pi_{\theta, \xi}^*, s_t)$ designed for different goals, therefore the objective function of the outer optimization is formulated as $\mathbb{E}[\sum_t f(\xi, \pi_{\theta, \xi}^*, s_t)]$. The evaluation of the fitness will be detailed in later subsection.

$$\xi^* = \arg \max_{\xi} \mathbb{E} \left[\sum_{t'} [f(\xi, \pi_{\theta, \xi}^*, s_{t'})] \right] \quad (2)$$

Then the objective of the bi-level optimization can be formulated as

$$\begin{aligned} \xi^*, \pi_{\theta, \xi}^* = \arg \max_{\xi} \mathbb{E} \left[\sum_{t'} [f(\xi, \pi_{\theta, \xi}^*, s_{t'})] \right], \\ \arg \max_{\pi_{\theta, \xi}} \mathbb{E} \left[\sum_t [\gamma^t r(s_t, a_t, \xi)] \right] \end{aligned} \quad (3)$$

3.2. Universal policy and pseudo-critic training

To evaluate the best performance of each design of humanoid robot, an optimal motion controller is needed, and model-free RL is a natural option. Our method involves the training of the following networks: universal policy $\pi(a|o, \xi)$ extended from $\pi(a|o)$, value function $V(s, \xi)$ and pseudo-critics $V_{pseudo}(s, \xi)$ extended from $V(s)$. The universal policy is trained together with pseudo-critics by Proximal Policy Optimization (PPO) algorithm [36].

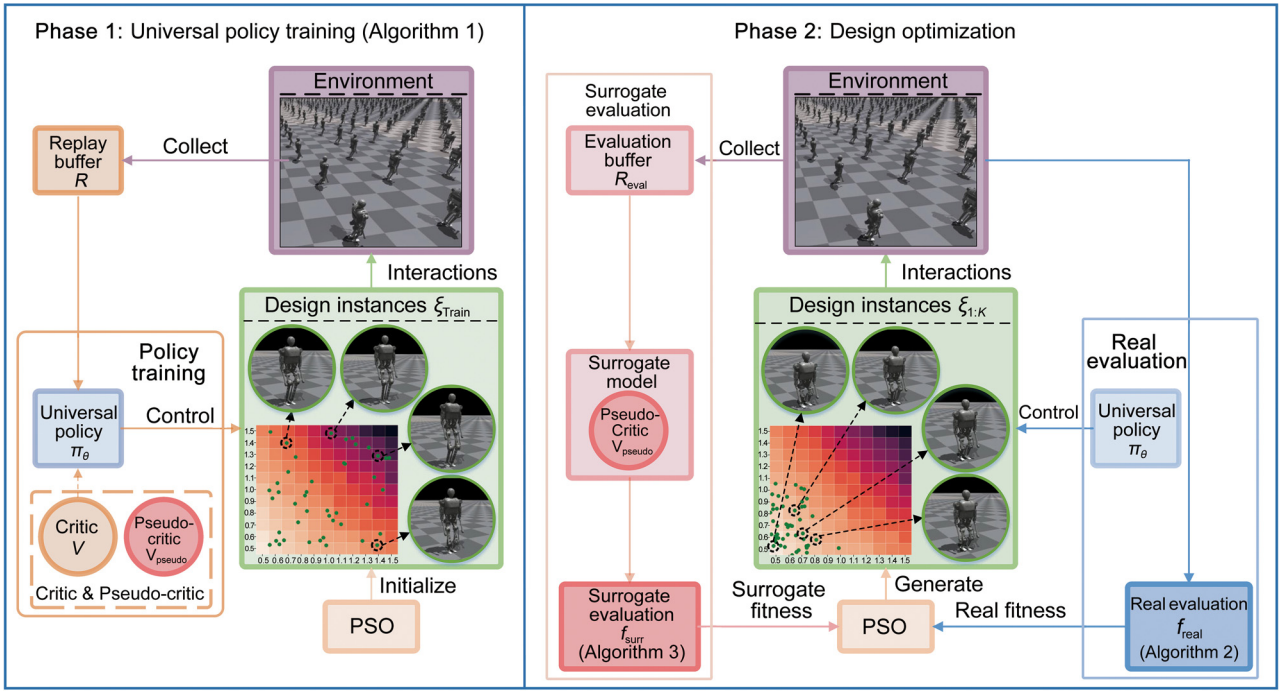


Fig. 2. Overview of the proposed design co-adaptation method. In phase 1, we train the universal policy and pseudo-critic concurrently with design instances uniformly sampled from current design space. In phase 2, we apply PSO to perform design optimization with both real fitness evaluation and surrogate fitness evaluation using pseudo-critic from phase 1. The two phases are conducted iteratively until convergence. In the figure, an example of how design instances are updated to low-cost area in design space is shown. The cost heat-map in design instances block shows the cost in design space, and lighter color corresponds to lower cost.

3.2.1. PolicyNet

The universal policy $\pi(a|o, \xi)$ accepts proprioception observation o_t and design parameters ξ as input, and infers the action a_t , enabling cross-morphology robots controlling. Observation $o_t \in \mathbb{R}^{70}$ contains base linear velocity, base angular velocity, base orientation, base velocity commands, joint position, joint velocity, action history, and clock for raibert heuristics.

The policy π_θ output $a_t \in \mathbb{R}^8$ as target positions for the 8 joints. Then the torques applied to the joints are mapped by a PD controller from a_t .

3.2.2. ValueNet & pseudo-critic

Normally, value network $V(s, \xi)$ functions as a component in the actor-critic paradigm which predicts the value of the current state. And it is also intuitive to employ it as fitness estimator by exploiting its capability of evaluating the state. However, the objective of the design optimization does not necessarily align with the objective of the RL training, so we propose to additionally train pseudo-critic networks $V_{pseudo}(s, \xi)$ as fitness estimator, which share architecture with $V(s, \xi)$ but are excluded from policy updates and predict task-aligned fitness metrics. We will show in experimental results section that training additional pseudo-critics reduces the total evaluation cost.

3.2.3. Reward shaping

Our reward setting includes command tracking reward for command (v_x, v_y, w_z) , torque penalty and other reward terms. Detailed reward terms can be found in Table 1. We apply heuristics to foot trajectory and corresponding reward to stabilize the learning process. For $x - y$ component of foot trajectory, we use Raibert heuristics [37] and von Mises distribution for z component.

The training process is shown in Algorithm 1. It is worth noting the universal policy is trained with design instances uniformly sampled from current range, and may not be optimal for each

Table 1

Reward functions.

Reward	Formula	Weight
Linear command tracking	$\exp(-4(v_{xy}^{cmd} - v_{xy})^2)$	2.0
Angular command tracking	$\exp(-4(\omega_{yaw}^{cmd} - \omega_{yaw})^2)$	1.5
Linear velocity (z)	v_z^2	-1.25
Angular velocity (xy)	ω_{yaw}^2	-0.1
Torque	τ^2	$-5.0 * 10^{-6}$
Orientation	$ g ^2$	-10.0
Joint acceleration	\ddot{q}	$-2.0 * 10^{-7}$
Action rate	$(a_t - a_{t-1})^2$	-0.025
Joint default position	$\sum_j (q_j - q_j^{stand})^2$	-0.2
Raibert heuristics	$\sum_j (q_j - q_j^*)^2$	-0.75

robot with specific morphology due to the limit of the network learning capacity. Therefore, we expands universal policy capacity via fine-tuning on narrowed design subspaces after the design optimization narrows the range of morphology, as shown later in Algorithm 4.

3.3. Surrogate-assisted design optimization

3.3.1. Fitness evaluation

Our framework employs a hybrid evaluation strategy, including real fitness evaluation $f_{real}(\xi, \pi_\theta)$ and surrogate fitness evaluation $f_{surr}(\xi, V_{pseudo})$. Real fitness evaluation calculates the fitness by Monte-Carlo estimation via environmental interactions, which can be formulated by

$$f_{real}(\xi, \pi_\theta) = \sum_{s_t \sim S_{\pi_\theta}} [P(s_t)] / |S_{\pi_\theta}| \quad (4)$$

Algorithm 1 Universal policy training

```

1: Input Universal policy  $\pi_\theta$ ,
2:   Multi-critics  $V = \{V_{critic}, V_{pseudo}\}$ ,
3:   Design instances  $\xi_{train}$ .
4: Initialize Replay buffer  $R$ ,
5:   Policy update epochs  $T_{policy}$ ,
6:   epoch length  $T$ ,
7:   Optimization metrics  $P$ .
8: for  $i = 1 : T_{policy}$  do
9:   for  $t = 1 : T$  do
10:    Collect tuple  $\{s_t, a_t, r_t, s_{t+1}\}$  with  $\pi_\theta(*, \xi_{train})$ 
11:    Insert tuple with  $P(s_t, a_t, s_{t+1})$ , then store in  $R$ 
12:   end for
13:   Update  $\pi_\theta$  and  $V$  with data in  $R$ 
14: end for

```

where P is the robot performance metric, S_{π_θ} is the set of robot states generated by policy π_θ , and $|S_{\pi_\theta}|$ is the volume of S_{π_θ} . The real evaluation process is shown in Algorithm 2.

The real evaluation requires a large amount of environmental experience. This problem can be significantly alleviated by batch evaluation of design instances using universal policy, but it is still relatively computational expensive to conduct frequent real evaluation. Instead, surrogate fitness evaluation can be applied to predict the fitness efficiently by averaging the predicted values over states, formulated by

$$f_{surr}(\xi, V_{pseudo}) = \sum_{s_t \sim S_{\pi_\theta}} [V(s_t)] / |S_{\pi_\theta}| \quad (5)$$

Surrogate evaluation reuses historical interaction data, so this process requires no extra computational cost. The surrogate evaluation process is shown in Algorithm 3.

Algorithm 2 Real fitness evaluation f_{real}

```

1: Input Design instances  $\xi_{1:K}$ ,
2:   Universal policy  $\pi_\theta$ .
3: Output Real fitness  $F_{real,1:K}$ , Evaluation buffer  $R_{eval}$ .
4: Initialize  $F_{real,1:K} = \{F_{real,1}, F_{real,2}, \dots, F_{real,K}\}$ ,
5:   Optimization metric  $P$ ,
6:   Evaluation length  $T_{eval}$ ,
7:   Evaluation buffer  $R_{eval}$ .
8: for  $t = 1 : T_{eval}$  do
9:   Collect  $\{s_{t,1:K}, a_{t,1:K}, r_{t,1:K}, s_{t+1,1:K}\}$  by  $\pi_\theta(*, \xi_{1:K})$ ,
10:  and store in  $R_{eval}$ 
11:  for  $k = 1 : K$  do
12:    $F_{real,k} = F_{real,k} + P(s_{t,k}, a_{t,k}, s_{t,k}, \xi_k) / T_{eval}$ 
13:  end for
14: end for
15: Return  $F_{real,1:K}, R_{eval}$ 

```

3.3.2. Design optimization

For design optimization, we apply gradient-free PSO considering the non-differentiable nature of our fitness evaluation method. PSO achieves global search by utilizing a swarm of particles that collaboratively explore the search space. This approach synergies with our framework's capacity for batch evaluation via the universal policy, enabling simultaneous assessment of particle which significantly increases efficiency.

Different metric functions can be applied for different optimization objectives. Similar to [6], we formalize three core objectives critical to humanoid locomotion design:

Algorithm 3 Surrogate fitness evaluation f_{surr}

```

1: Input Design instances  $\xi_{1:K}$ ,
2:   Pseudo-critic  $V_{pseudo}$ ,
3:   Evaluation buffer  $R_{eval}$ .
4: Output Surrogate fitness  $F_{surr,1:K}$ 
5: Initialize  $F_{surr,1:K} = \{F_{surr,1}, F_{surr,2}, \dots, F_{surr,K}\}$ ,
6:   Evaluation length  $T_{eval}$ ,
7: Sample  $T_{eval}$  states from  $R_{eval}$ 
8: for  $t = 1 : T_{eval}$  do
9:   for  $k = 1 : K$  do
10:     $F_{surr,k} = F_{surr,k} + V_{pseudo}(s_t, a_t, s_t, \xi_k) / T_{eval}$ 
11:   end for
12: end for
13: Return  $F_{surr,1:K}$ 

```

Algorithm 4 Design co-adaptation

```

1: Initialize Optimization iterations  $N$ ,
2:   Particle population  $K$ , PSO iterations  $J$ ,
3:   Evaluation probability  $w$ ,
4:   Init parameter distribution  $\mathcal{E}_1 \sim [x_l, x_u]$ ,
5:   Random number generator  $G$ .
6: Initialize Universal policy  $\pi_\theta$ ;
7:   Multi-critics  $V = \{V_{critic}, V_{pseudo}\}$ ,
8:   Evaluation buffer  $R_{eval}$ .
9: for  $n = 1 : N$  do
10:  Sample  $K$  design instances  $\xi_{1:K} \sim \mathcal{E}_n$ 
11:  UniversalPolicyTraining( $\pi_\theta, V, \xi_{1:K}$ ) ▷ Algorithm 1
12:
13:  for  $j = 1 : J$  do
14:   if  $G > w$  then
15:     $F_{real,1:K}, R_{eval} = f_{real}(\xi_{1:K}, \pi_\theta)$  ▷ Algorithm 2
16:   else
17:     $F_{surr,1:K} = f_{surr}(\xi_{1:K}, V_{pseudo}, R_{eval})$  ▷ Algorithm 3
18:   end if
19:
20:    $\xi_{1:K,j+1} = Update(\xi_{1:K,j}, F_{1:K})$ 
21:  end for
22:   $\mathcal{E}_{n+1} = Update(\xi_{1:K,j})$ 
23: end for

```

- Command tracking:

$$P_{cmd} = e_{v,t}^2 + e_{\omega,t}^2 \quad (6)$$

- Power consumption reduction:

$$P_p = \omega_t \sum_{i \in \{1, \dots, 8\}} |\tau_{i,t}| |\dot{\theta}_{i,t}| \quad (7)$$

- Joint torque reduction:

$$P_\tau = \omega_t \sum_{i \in \{1, \dots, 8\}} \tau_{i,t}^2 \quad (8)$$

where e_v is linear velocity command tracking error, e_ω is angular velocity command tracking error, τ is joint torque, $\dot{\theta}$ is joint velocity, and $\omega_t = \exp(e_{v,t}^2 + e_{\omega,t}^2)$ which serves as an adaptive penalty term to avoid convergence to static local optima. These fitness functions we selected represent the key challenges in humanoid robot design for locomotion: P_{cmd} ensures stable command tracking under velocity commands. P_p and P_τ mitigate the energy waste and mechanical stress, which are critical in real-world robot design. When designers seek to find an optimal design of robot, there is always trade-off among the three or

more objectives. So a practical objective could be a combination of the objectives according to the application scenario. So in our experiment we choose the representative ones to show the capability of our framework to optimize the robot design with potential objectives.

While the surrogate model enables rapid fitness estimation, its inherent approximation errors risk suboptimal design convergence. To mitigate this, we implement a probabilistic hybrid evaluation: During optimization, either of the two evaluation methods is used randomly at a fixed evaluation probability w . Then we can combine the efficient while potentially inaccurate surrogate evaluation and costly but accurate real evaluation to achieve a balance between performance and efficiency for fitness evaluation.

The overview of the co-adaptation algorithm is outlined in Algorithm 4. Each iteration of the bi-level optimization framework initiates by sampling design instances from the dynamically narrowed design space. These instances are used jointly to train the universal policy and pseudo-critics, enabling the policy to generalize across diverse morphologies while the pseudo-critics approximate fitness metrics. The PSO then employs a hybrid evaluation strategy, which combines real environmental interactions for precision and surrogate predictions from pseudo-critics for efficiency, to update design candidates toward high-fitness regions. Finally, the design space is refined by narrowing parameter ranges around optimized particles, prioritizing high-performance regions for subsequent iterations. This cycle iterates until convergence, combining computational efficiency and accuracy through adaptive sampling and evaluation.

4. Experiment and results

In this section, by providing the implementation details and results of our experiment, we aim to evaluate our framework's ability to: 1. enable computationally efficient design-control co-adaptation. 2. outperform baseline methods in balancing optimization fidelity and resource efficiency?

4.1. Experimental setup

4.1.1. Kuafu humanoid robot design

In this paper we use self-developed Kuafu humanoid robot as experimental platform. The Kuafu humanoid robot weighs 42.0 kg. The height of Kuafu is 1.6 m, and the default length of the thigh l_0 and shank l_1 are both 0.35 m. It has 10 DoFs with 2 DoFs on shoulder and 8 DoFs on legs, and each leg contains abductor/adduction, hip, knee, ankle joint. We fix the both shoulder DoFs during optimization since they are irrelevant to our experiment tasks. The Kuafu humanoid robot in different environments are shown in Fig. 1.

In this paper, we aim to optimize the kinematics of the robot, including the length of thigh and shank. We parameterize the design $\xi \in \mathbb{R}^2$ as scaling factor of leg length and range $\xi \in [0.5, 1.5]^2$, and therefore the length of legs are $l_0 = \xi_0 \times 0.35$, $l_1 = \xi_1 \times 0.35$.

4.1.2. Morphology generation

To generate diverse design instances for the optimization framework, we adapt the humanoid robot model by proportionally scaling its leg length according to each candidate design parameterization. Following geometric scaling, corresponding adjustments to mass and inertia properties are implemented. Instead of using simplified rectangular boxes model as leg components in [38], we derive mass and inertia parameters through polynomial regression models calibrated against empirical data extracted from SolidWorks CAD models. Proportional-derivative

Table 2
Domain randomization setting.

Parameter	Range	Unit
Friction coefficient	[0.5, 1.25]	N/m ²
Restitution	[0.0, 0.4]	–
Base CoM shift	[–2.5, 2.5]	cm
PD factor	[0.9, 1.1]	–
Motor strength	[0.9, 1.1]	–
System delay	[0, 20]	ms

Table 3
Noise level.

Parameter	scale	Unit
Joint position	0.01	rad
Joint velocity	1.5	rad/s
Base linear velocity	0.01	m/s
Base angular velocity	0.2	rad/s
Projected gravity	0.05	rad/s ²

(PD) control gains are systematically adjusted in accordance with inertial properties, adhering to the physical principle that increased limb mass necessitates higher actuator forces to achieve equivalent dynamic performance.

4.1.3. Implementation details

The training environments are implemented using Isaac Gym, a PhysX engine based simulator that is tailored for massive parallel training with GPU acceleration [39]. Our code implementation is extensively based on [40]. We model the control policy as 3-layer Multi Layer Perception (MLP) whose hidden size is [512, 256, 128]. We use MLP instead of Transformer as in [15] for its training is much more computationally efficient. Both the critic and pseudo-critic are also parameterized by 3-layer MLP whose hidden size is [512, 256, 128]. The training is performed on a desktop PC with an Intel Core i7-13700k CPU @ 3.40 GHz, 128 GB RAM, and an NVIDIA RTX 4090 GPU.

To make sure the performance of the controller after transfer, domain randomization and noise are applied during policy training. The domain randomization settings and noise settings are shown in Tables 2 and 3 respectively.

The PSO algorithm is based on the implementation of Py-moo [41], with the following parameterization: optimization iteration $N = 3$, particle population $K = 100$, each corresponding to a design instance, PSO iteration $J = 5$, momentum term $w = 0.9$, cognitive impact $c1 = 2.0$, social impact $c2 = 2.0$, control factor $\omega = 0.5$. During both retraining and real fitness evaluation, we set 50 agents for each design instance. During experiment, we give randomly sampled command $v^{cmd} \sim U(-0.5, 1.25)$.

4.2. Effect of universal policy

First we need to justify the usage of universal policy. We compare the performance of universal policy with specialized policy in both rewards and training cost. We show the training curves of universal policy trained with designs ξ_{Train} sampled from (1) range [0.5, 1.5]², (2) range [0.65, 1.35]², (3) range [0.75, 1.25]² and (4) specialized policy, in Fig. 3. Note that the universal policies are trained with $K = 100$ design instances. Also, to evaluate the robustness of universal policy on the extreme designs, the design instance [0.5, 0.5] and [1.5, 1.5] are further tested with similar settings. The results of rewards are shown in Tables 4 and 5. The results reveal that all policies exhibit comparable convergence rates, requiring similar numbers of training iterations. Performance degradation scales with parametric domain breadth, universal policies achieve similar performance to specialized policy when the parametric domain is sufficiently constrained. This

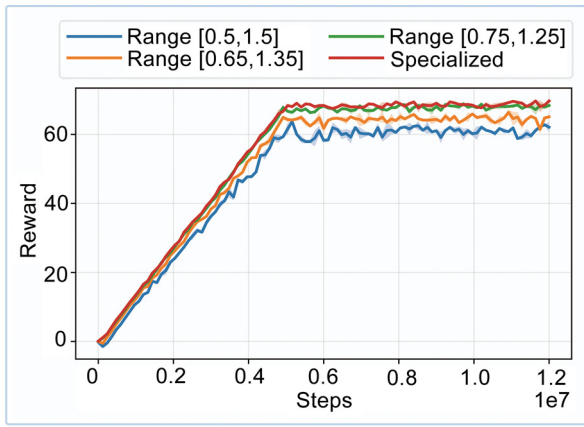


Fig. 3. Curve of rewards during training. Note that the rewards are evaluated only using the original morphology. As the morphology range narrows down, the rewards increase and finally are comparable to specialized policy.

Table 4

Design instance [0.5, 0.5] rewards.

Specialized Policy	Range [0.5, 1.5] ²	Range [0.5, 1.0] ²	Range [0.5, 0.75] ²
58.4 ± 1.87	54.2 ± 2.10	57.1 ± 1.95	57.9 ± 1.69

Table 5

Design instance [1.5, 1.5] rewards.

Specialized Policy	Range [0.5, 1.5] ²	Range [1.0, 1.5] ²	Range [1.25, 1.5] ²
60.8 ± 1.58	56.5 ± 1.76	58.4 ± 1.45	60.0 ± 1.20

demonstrates that universal policy implementation achieves substantial reductions in computational intensity while maintaining acceptable performance levels. Furthermore, the integrated iterative optimization framework, which retrains the universal policy at each optimization cycle, ensures continuous policy adaptation, preventing stagnation in local optima during design space exploration.

4.3. Effect of surrogate model

We further assess the efficacy of the pseudo-critic as a fitness estimation mechanism and quantify its associated computational cost. Fig. 4 presents comparative training trajectories and fitness prediction errors across three pseudo-critic variants optimized for distinct representative objectives. The results demonstrate progressive improvement in estimation accuracy during training iterations, while the trained pseudo-critics still produce larger-than-expected and unevenly distributed errors in evaluation. It is worth noting that the source of the unevenly distributed errors could be the variance of the objective fitness. Though the humanoid robot locomotion for which we aim to optimize the design and control is periodically repeated motion, the prediction of the value of its design objectives fitness function remains a task with high variance. Its value can be easily affected by the noise, perturbations, random seed etc. The integration of this auxiliary network architecture incurs a modest 3% increase in computational time. These findings suggest that while the pseudo-critic surrogate model provides computationally efficient guidance for design space exploration, its uncorrected use still risks convergence to local optima.

4.4. Design optimization of kuafu robot

We validate our framework through co-adaptation experiments on a humanoid robot performing locomotion across three

terrain types: Flat, Slope, Random. The example terrains are shown in Fig. 5. And we adopt the straightforward terrain generation method. The slope terrain is generated by changing the normal vector of the plane, to get a 10-degree slope. The random terrain is generated by uniformly sampling the height of terrain to get a uniform noise terrain. For better evaluation of the performance and efficiency of the proposed method, we compare our method using the same objectives introduced previously with the following algorithms:

Ours w/o Surrogate Evaluation: Surrogate fitness evaluation is excluded during optimization.

Ours w/o Real Evaluation: Real fitness evaluation is excluded during optimization.

Ours w/o Policy Retraining: The universal policy is not retrained in each iteration of optimization. And only real fitness evaluation is applied.

Bayesian Optimization & Specialized Policy: Specialized policy is used instead of universal policy in design optimization framework. And the sample-efficient Bayesian Optimization is applied since the training and evaluation of the instances sampled by PSO can be computationally intractable. This method will be denoted as BO in later sections.

4.4.1. Optimization result

The results of the design co-adaptation are shown in Table 6 and partly as cost heat-map in Fig. 6. It is worth noting for the improvement item in table, we calculate the improvement of command tracking objective by comparing the tracking error. While the improvement of power and torque reduction are calculated directly by cost reduction considering the computation of both objectives are coupled with the command tracking performance.

On flat terrain, the results suggest, with relatively large command range ($U(-0.5, 1.25)$), the design tends to have longer thigh and shank with the objective of command tracking and power reduction. On slope and random terrains, the designs opt for longer thigh and shorter shank to overcome the terrains, while the overall leg length is still increased. For torque reduction objective, designs in all terrains tend to have minimal leg length to reduce the mass and inertia and therefore result in significant drop in torque. In slope and random terrains, the leg length is slightly higher to provide necessary height to overcome terrains.

The results show *ours w/o surrogate fitness evaluation* outperforms other methods in most experiments. The proposed method achieves comparable performance to *ours w/o surrogate evaluation*, and outperforms other baseline methods including *ours w/o real fitness evaluation* and *ours w/o policy retraining*. The results of *ours w/o policy retraining* suggest the initial universal policy trained on a broad morphology range fails to adapt to specialized morphologies, and introduces bias in fitness evaluation. And the results of *ours w/o real fitness evaluation* suggest the uneven errors of surrogate models can also lead to suboptimal convergence. But this can be effectively corrected by the design space narrowing and occasional real evaluation. Fig. 6 shows the cost heat-map corresponding to the three objectives in flat terrain. It clearly suggests the proposed method and *ours w/o surrogate fitness evaluation* can finally reach the global optimum, while other baseline methods fail the optimization.

4.4.2. Sim-to-sim validation

The ultimate goal of the humanoid robot co-adaptation is to flexibly change the robot design according to different objectives and transfer to different environments, especially in real world. So we evaluate the transfer performance of optimized designs in a sim-to-sim experiment, in which we transfer the optimized design and controller to another simulated environment. Because

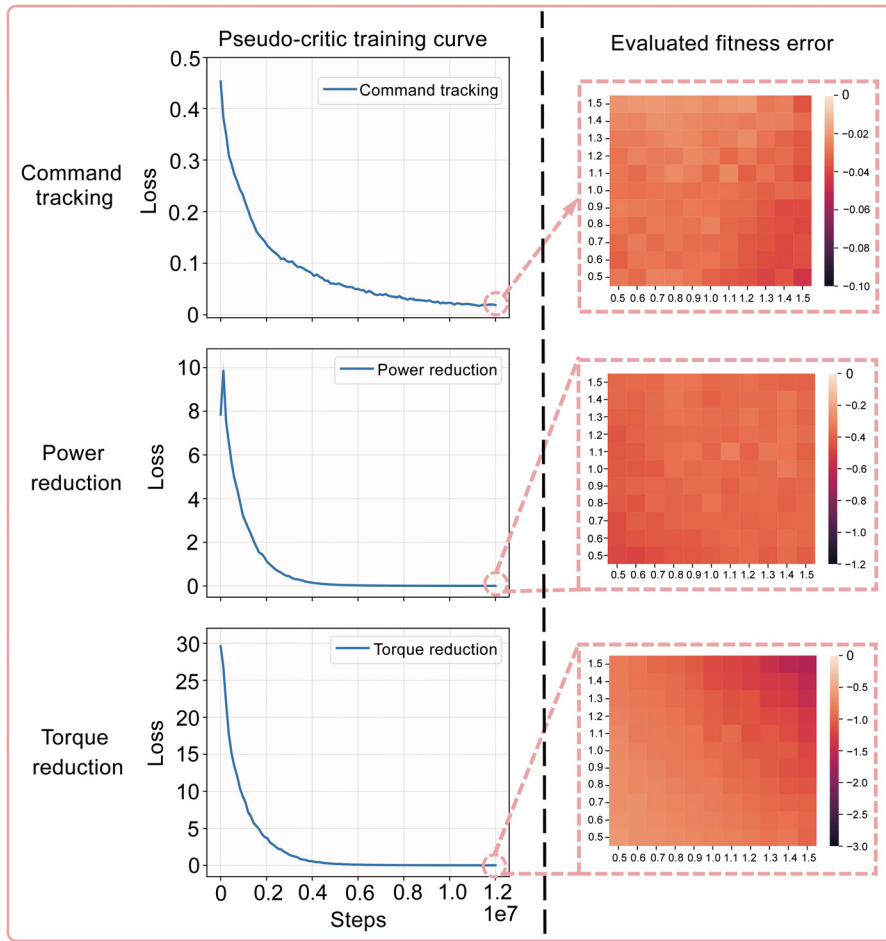


Fig. 4. The training and evaluation results of the pseudo-critic using the three representative objectives. From top to bottom, they are: (1) command tracking, (2) power reduction, (3) torque reduction. The training process is shown in the left-hand side. And the evaluation errors in design space of the trained pseudo-critic is shown as heat-maps in the right-hand side. The evaluation is performed by calculating the error between the predicted fitness using trained pseudo-critics and the real fitness. The results suggest the unevenly distributed evaluation error still exists after the training converged. The mean errors are 0.029, 0.386, 0.946 respectively.

Table 6
Optimization results.

Objective	Method	Flat				Slope				Random			
		High	Shank	Improvement	Sim-to-sim Improvement	High	Shank	Improvement	Sim-to-sim Improvement	High	Shank	Improvement	Sim-to-sim Improvement
Cmd	Ours	1.41	1.13	5.8%	6.6%	1.23	1.33	5.1%	4.5%	1.36	0.86	4.6%	5.8%
	Ours w/o Surrogate	1.38	1.09	7.2%	6.5%	1.21	1.25	3.3%	2.9%	1.35	0.88	5.4%	5.9%
	Ours w/o Real	1.45	1.12	5.7%	6.9%	1.08	1.36	4.3%	2.7%	1.23	1.00	3.1%	4.0%
	Ours w/o Retraining	1.47	0.97	3.0%	3.2%	1.02	1.10	1.2%	1.4%	0.77	1.00	0.2%	-4.4%
	BO	1.08	0.98	1.5%	1.7%	0.94	0.92	-1.0%	-0.7%	1.35	1.30	-3.9%	-1.2%
P	Ours	1.36	1.40	11.0%	8.0%	1.40	0.91	8.1%	4.7%	1.20	0.90	13.0%	11.7%
	Ours w/o Surrogate	1.39	1.39	11.3%	8.4%	1.43	0.95	9.7%	5.4%	1.21	0.90	12.9%	11.4%
	Ours w/o Real	1.26	1.27	4.2%	4.3%	1.25	0.88	4.8%	2.5%	1.26	0.85	12.1%	10.7%
	Ours w/o Retraining	1.40	1.08	8.5%	6.3%	1.29	1.04	-0.5%	-1.1%	1.21	1.00	7.5%	5.6%
	BO	0.96	1.49	1.0%	1.8%	1.06	1.50	-3.6%	-3.6%	1.25	1.45	3.6%	-2.0%
τ	Ours	0.51	0.54	45.9%	28.0%	0.56	0.55	25.6%	16.0%	0.58	0.51	27.0%	13.6%
	Ours w/o Surrogate	0.50	0.51	46.5%	28.4%	0.55	0.54	25.7%	15.8%	0.58	0.51	27.0%	13.6%
	Ours w/o Real	0.53	0.57	43.6%	26.8%	0.56	0.53	24.5%	15.9%	0.63	0.51	26.8%	14.4%
	Ours w/o Retraining	0.50	0.50	46.8%	28.9%	0.50	0.51	25.1%	10.2%	0.53	0.54	26.8%	10.9%
	BO	0.50	0.50	46.8%	28.9%	0.50	0.50	25.1%	10.6%	0.50	0.50	26.6%	8.5%

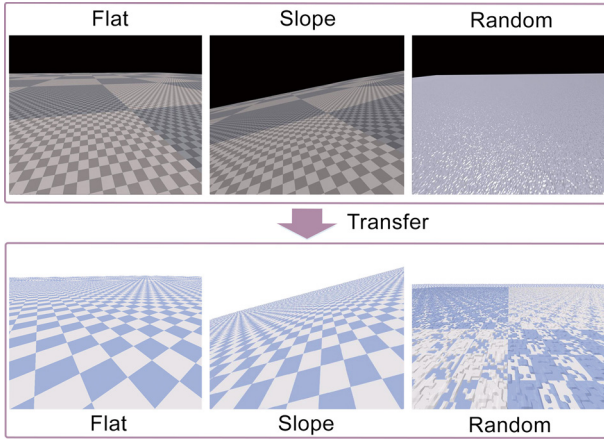
different simulated environments differ in physical properties, the experiment can be a proof of the effectiveness of the proposed method. With this goal, we use Pybullet, a bullet-engine based simulation as target environment. The results of the sim-to-sim experiments are also shown in Table 6, and is consistent with the previous results in general. Note that in the experiment with objective of torque reduction, the overall improvement is lower due to the worse command tracking performance after transfer with short legs.

4.4.3. Efficiency analysis

We then analyze the computational efficiency of each method, which is evaluated from the environmental interactions and wall clock time consumption. The evaluation is conducted by both the aspects because the number of environmental interactions can be accurately calculated while the consumed time can be affected by other factors and does not necessarily grows linearly. So we also evaluate the consumed wall clock time. The number of expected environmental interactions and average wall clock time consumption, and their constituent parts including training and

Table 7
Computational cost.

Method	Interactions (million, Training+Evaluation)	Wall clock time (hour, Training+Evaluation)
Ours	102(72+30)	1.07(0.4+0.67)
Ours w/o surrogate	132(72+60)	1.73(0.4+1.33)
Ours w/o real	72(72+0)	0.4(0.4+0)
Ours w/o retraining	60(0+60)	1.33(0+1.33)
BO	200(200+0)	3.0(2.67+0.33)

**Fig. 5.** Examples terrains. The top part of the figure shows example terrains in Isaac Gym, while the bottom part shows the corresponding terrains in PyBullet. Note that we use a 10° slope for slope terrain, and random obstacles with maximum height of 5 cm for random terrain.

evaluation, are shown in Table 7. The results suggest, resulted from other components in the program including morphology generation etc., the evaluation is more costly than training, despite it requires less environmental interactions. Note that the BO baseline only evaluates 20 potential designs in 3 h, for comparison the proposed method evaluates 1500 designs in about 1 h. The computational inefficiency of specialized policy and BO explains the previous sub-optimal results and proves the usage of universal policy can significantly shorten the training and evaluation time. From the results, we can verify that our method can effectively reduce the computational cost of evaluation, and thereby the total cost of design optimization.

Overall, the proposed method balances computational efficiency and optimization accuracy through two key mechanisms: Universal policy with iterative design space narrowing and hybrid fitness evaluation. Compared to baselines, BO with specialized policies is inefficient due to its sequential sampling. It evaluates only 20 designs, which is insufficient for high-dimensional spaces. Ours w/o Retraining remove iterative narrowing and retraining, and therefore reduces training time but degrades performance due to suboptimal controllers. Ours w/o Real excludes real evaluations and saves 60% cost but suffers from surrogate prediction errors. Ours w/o Surrogate uses pure real evaluation, and therefore achieves marginally higher accuracy but incurs 60% higher computational cost than ours. So we can draw the conclusion that the hybrid evaluation strategy retains accuracy at lower evaluation cost. And the iterative narrowing improves specialization. Though retraining adds training time, it ensures robust convergence. The framework outperforms baselines in efficiency and adaptability, validated across terrains. Therefore, the proposed method reaches the balance of accuracy and efficiency.

4.5. Mixed-variable design optimization of kuafu robot

In real robot design problem, there are also discrete variables in design parameters such as motor types, robot body materials

Table 8

Motor types.

Motor type	Torque limit (NM)	Speed limit (RPM)	Motor mass (kg)
1	119.7	99	0.62
2	120.3	340	0.62
3	135.1	310	0.67

Table 9

Mixed-variable optimization results.

Objective	Terrain	Thigh	Hip motor	Shank	Knee motor	Improvement
V	Flat	1.41	1	1.08	1	6.4%
	Slope	1.24	2	1.36	3	4.4%
	Random	1.33	1	0.85	2	4.9%
P	Flat	1.41	1	1.33	2	11.4%
	Slope	1.43	1	0.95	2	12.3%
	Random	1.23	2	0.92	3	12%
T	Flat	0.54	3	0.53	2	33.5%
	Slope	0.61	2	0.51	3	22.4%
	Random	0.58	2	0.53	2	24.8%

etc. To further demonstrate the ability of our method to address real robot design problem, we extend our framework to handle hybrid design spaces $\xi \in \mathbb{R}^2 \times \mathbb{D}^2$, by including motor types of hip and knee in design parameter as discrete variable.

The potential motor types and parameters are listed in Table 8. The torque and speed limit of different motors are considered in the robot model. We choose these motor types because they have been used for hip and knee joint in the previous humanoid robot designs. The type 1 in table is the motor currently used in Kuafu robot. Note that the gear ratio of both the hip and knee are 17.43 and are considered in Table 8.

We use the same previous optimization setting, and the results are shown in Table 9. The motor type mostly functions as constraint on torque and speed. The results show the motor 1 is never used in the optimization with torque objective due to its low speed limit. Other than that, the type of motor does not affect the results considering the difference of motors in mass is negligible and all three motors can meet the torque requirement. The hybrid optimization achieves comparable performance gains to the continuous-only case, demonstrating the framework's capacity to handle mixed-variable spaces without performance degradation. This demonstrates that when actuator mass differences are small, choosing between off-the-shelf motors mainly affects whether the design meets physical limits (like torque/speed) rather than directly improving performance goals, which is a crucial advantage for real applications where engineers must use standard components.

It is worth noting that the dimension of parameters is still relatively low comparing to the real-world humanoid robot design problem. As the dimensionality of the design space increases, the computational cost and numbers of the evaluation to maintain the similar coverage of design space grows exponentially. The pseudo-critic prediction errors will also grow, as the training data becomes sparse relative to the expanded design space. To mitigate the challenge and ensure tractable optimization, the high-dimensional spaces can be projected into lower-dimensional manifolds, such as PCA, autoencoders, to retain performance-critical features. And applying the uncertainty-aware sampling

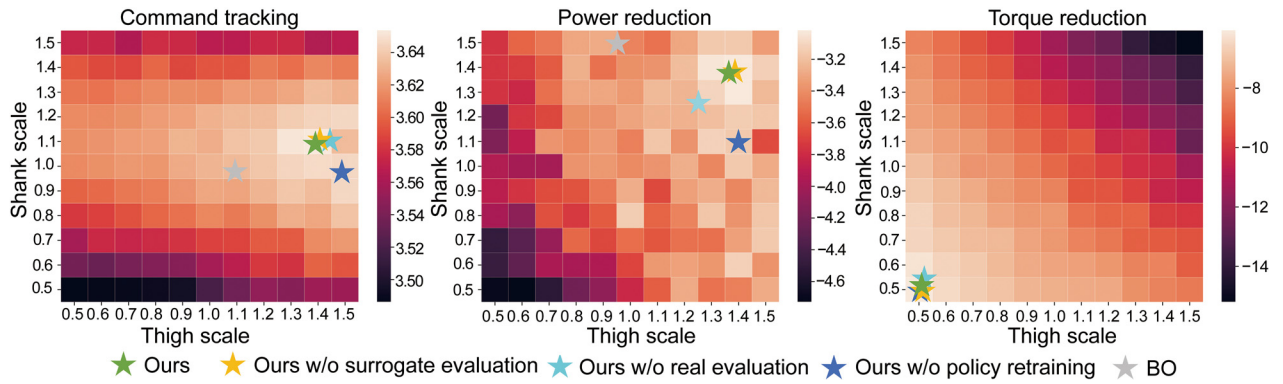


Fig. 6. Cost heat-map of different optimization objectives on flat terrain. From left to right the objectives are 1. command tracking, 2. power reduction, 3. torque reduction, respectively. In the heat-map, lighter color corresponds to lower cost. We can see from results both *ours w/o surrogate evaluation* and *ours* could find the global optimum in all scenarios, while others failed in at least one experiment.

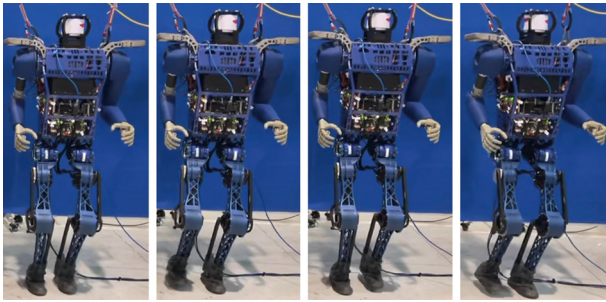


Fig. 7. Snapshots of Kuafu robot experiment.

Table 10

Objective cost values.

Isaac gym	Pybullet	Real robot
-8.77 ± 0.15	-9.01 ± 0.17	-9.15 ± 0.17

strategy which prioritizes evaluations in high-uncertainty regions using ensemble-based variance detection can address the potential prediction error issue.

4.6. Real robot validation

Due to the high cost of building a robot of different morphologies, currently we are only able to validate the objective cost values using the existing Kuafu robot ($\xi = [1.0, 1.0, \text{type1}, \text{type1}]$). Here we comprehensively consider the optimization objective and assume the objective to be the $P_{real} = P_{cmd} + P_p + P_r$. The snapshots of experiment are provided in Fig. 7, and the corresponding values are shown in Table 10. The results indicate that objective values exhibit marginal deviation between simulated and real-world environments. The observed cost degradation in PyBullet and on the real robot is primarily due to higher torque requirements. While these experiments do not constitute direct real-world validation of the previous results, they provide a preliminary verification that the results remain consistent with prior benchmarks, suggesting limited divergence.

5. Conclusion

In this paper, we present a novel DRL-based framework for efficient humanoid robot design co-adaptation. We formulate the design co-adaptation problem as a bi-level optimization problem. In the inner loop, we train universal policy and pseudo-critic,

which is prepared as surrogate model, on robots of diverse morphologies. Then in the outer loop, PSO is applied with a combination of real fitness evaluation and surrogate fitness evaluation. Experiments with continuous and mixed variables in simulation are conducted. Results suggest the application of universal policy and surrogate model in the proposed method improves the capability of finding the optimal designs of humanoid robot with respect to different objectives and reduces the computational cost. And the proposed framework can also be easily extended to the co-adaptation of other robots with any objective considering all components of the framework are model-free.

The limitations of this paper are primarily twofold. Firstly, due to high cost of building robot, we were unable to build a prototype based on the optimization results to validate our method by sim-to-real experiment. And the sim-to-sim experiment cannot fully demonstrate the effectiveness of the proposed method in real-world robot co-adaptation problem. Secondly, the dimension of the parameters we optimized was still low, whereas real-world humanoid robot design problems typically involve optimization of a larger number of parameters. Therefore, in future work, we should explore ways to build low-cost prototypes to validate our method, and identify more reasonable parameters for optimization.

CRedit authorship contribution statement

Yidong Du: Writing – review & editing, Writing – original draft, Visualization, Validation, Software, Methodology, Conceptualization. **Xuechao Chen:** Writing – review & editing, Funding acquisition. **Zhangguo Yu:** Writing – review & editing. **Fei Meng:** Writing – review & editing. **Zishun Zhou:** Writing – review & editing. **Yuanxi Zhang:** Writing – review & editing. **Qingqing Li:** Writing – review & editing. **Qiang Huang:** Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (62403060), in part by the Postdoctoral Fellowship Program of CPSF (GZC20233399), in part by the Beijing Natural Science Foundation (L243004), and in part by the “111” Project (B08043).

Appendix A. Supplementary data

Supplementary material related to this article can be found online at <https://doi.org/10.1016/j.birob.2025.100255>.

References

- [1] C. Dong, Z. Yu, X. Chen, H. Chen, Y. Huang, Q. Huang, Adaptability control towards complex ground based on fuzzy logic for humanoid robots, *IEEE Trans. Fuzzy Syst.* 30 (6) (2022) 1574–1584.
- [2] H. Chen, X. Chen, C. Dong, Z. Yu, Q. Huang, Online running pattern generation for humanoid robot with direct collocation of reference-tracking dynamics, *IEEE/ASME Trans. Mechatron.* 29 (3) (2024) 2091–2102.
- [3] I. Radosavovic, T. Xiao, B. Zhang, T. Darrell, J. Malik, K. Sreenath, Real-world humanoid locomotion with reinforcement learning, *Sci. Robot.* 9 (89) (2024) eadi9579.
- [4] Z. Fu, Q. Zhao, Q. Wu, G. Wetzstein, C. Finn, HumanPlus: Humanoid shadowing and imitation from humans, in: 8th Annual Conference on Robot Learning, 2024.
- [5] C. Dong, X. Chen, Z. Yu, H. Liu, F. Meng, Huang, Q., Swift running robot leg: Mechanism design and motion-guided optimization, *IEEE/ASME Trans. Mechatron.* 29 (3) (2023) 1702–1713.
- [6] Á. Belmonte-Baeza, J. Lee, G. Valsecchi, M. Hutter, Meta reinforcement learning for optimal design of legged robots, *IEEE Robot. Autom. Lett.* 7 (4) (2022) 12134–12141.
- [7] F. Bjelonic, J. Lee, P. Arm, D. Sako, D. Tateo, J. Peters, M. Hutter, Learning-based design and control for quadrupedal robots with parallel-elastic actuators, *IEEE Robot. Autom. Lett.* 8 (3) (2023) 1611–1618.
- [8] K.S. Luck, R. Calandra, M. Mistry, What robot do I need? Fast co-adaptation of morphology and control using graph neural networks, 2021, arXiv preprint arXiv:2111.02371.
- [9] M. Chadwick, H. Kolvenbach, F. Dubois, H.F. Lau, M. Hutter, Vitruvius: An open-source leg design optimization toolbox for walking robots, *IEEE Robot. Autom. Lett.* 5 (4) (2020) 6318–6325.
- [10] S. Hu, Z. Yang, G. Mori, Neural fidelity warping for efficient robot morphology design, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, 2021, pp. 7079–7086.
- [11] X. Zhu, P. Gergondet, Z. Cai, X. Chen, Z. Yu, Q. Huang, A. Kheddar, The development of a 7-dof humanoid arm for driving using a task-driven design method, *IEEE/ASME Trans. Mechatron.* 29 (2) (2023) 1521–1533.
- [12] K.M. Digumarti, C. Gehring, S. Coros, J. Hwangbo, R. Siegwart, Concurrent optimization of mechanical design and locomotion control of a legged robot, *Mob. Serv. Robot.* (2014) 315–323.
- [13] S. Ha, S. Coros, A. Alspach, J. Kim, K. Yamane, Task-based limb optimization for legged robots, in: 2016 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS, 2016, pp. 2062–2068, 2014.
- [14] J. Won, J. Lee, Learning body shape variation in physics-based characters, *ACM Trans. Graph.* 38 (6) (2019) 1–12.
- [15] A. Gupta, L. Fan, S. Ganguli, et al., Metamorph: learning universal controllers with transformers, 2022, arXiv preprint arXiv:2203.11931.
- [16] H. Tong, C. Huang, L.L. Minku, X. Yao, Surrogate models in evolutionary single-objective optimization: A new taxonomy and experimental study, *Inf. Sci.* 562 (2021) 414–437.
- [17] Y. Wang, T. Zhang, Y. Chang, X. Wang, B. Liang, B. Yuan, A surrogate-assisted controller for expensive evolutionary reinforcement learning, *Inf. Sci.* 616 (2022) 539–557.
- [18] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, M. Hutter, Learning agile and dynamic motor skills for legged robots, *Sci. Robot.* 4 (2019) 26.
- [19] J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, M. Hutter, Learning quadrupedal locomotion over challenging terrain, *Sci. Robot.* 5 (47) (2020) eabc5986.
- [20] T. Miki, J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, M. Hutter, Learning robust perceptive locomotion for quadrupedal robots in the wild, *Sci. Robot.* 7 (62) (2022) eabk2822.
- [21] F. Jenelten, J. He, F. Farshidian, Hutter, M., Dtc: Deep tracking control, *Sci. Robot.* 9 (86) (2024) eadh5401.
- [22] J. Siekmann, S. Valluri, J. Dao, F. Bermillo, H. Duan, A. Fern, Hurst, J., Learning memory-based control for human-scale bipedal locomotion, in: *Robotics: Science and Systems XVI*, 2020.
- [23] J. Siekmann, Y. Godse, A. Fern, J. Hurst, Sim-to-real learning of all common bipedal gaits via periodic reward composition, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, 2021, pp. 7309–7315.
- [24] Z. Li, X.B. Peng, P. Abbeel, S. Levine, G. Berseth, K. Sreenath, Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control, *Int. J. Robot. Res.* (2024) 02783649241285161.
- [25] T. Haarnoja, B. Moran, G. Lever, S.H. Huang, D. Tirumala, J. Humplik, N. Heess, Learning agile soccer skills for a bipedal robot with deep reinforcement learning, *Sci. Robot.* 9 (89) (2024) eadi8022.
- [26] A. Tang, T. Hiraoka, N. Hiraoka, F. Shi, K. Kawaharazuka, K. Kojima, M. Inaba, Humanmimic: Learning natural locomotion and transitions for humanoid robot via wasserstein adversarial imitation, in: 2024 IEEE International Conference on Robotics and Automation, ICRA, 2024, pp. 13107–13114.
- [27] G. Fadini, T. Flayols, A. Del Prete, N. Mansard, P. Souères, Computational design of energy-efficient legged robots: Optimizing for size and actuators, in: 2021 IEEE International Conference on Robotics and Automation, ICRA, 2021, pp. 9898–9904.
- [28] Y. Sun, C. Zong, F. Pancheri, T. Chen, T.C. Lueth, Design of topology optimized compliant legs for bio-inspired quadruped robots, *Sci. Rep.* 13 (1) (2023) 4875.
- [29] J. Xu, T. Chen, L. Zlokapa, M. Foshey, W. Matusik, S. Sueda, P. Agrawal, An end-to-end differentiable framework for contact-aware robot design, in: *Robotics: Science and Systems*, 2021.
- [30] D. Ha, Reinforcement learning for improving agent design, *Artif. Life* 25 (4) (2019) 352–365.
- [31] K.S. Luck, H.B. Amor, R. Calandra, Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning, in: *Conference on Robot Learning*, 2020, pp. 854–869.
- [32] C. Schaff, D. Yunis, A. Chakrabarti, M.R. Walter, Jointly learning to construct and control agents using deep reinforcement learning, in: 2019 International Conference on Robotics and Automation, ICRA, 2019, pp. 9798–9805.
- [33] A. Zhao, J. Xu, M. Konaković-Luković, J. Hughes, A. Spielberg, D. Rus, W. Matusik, Robogrammar: Graph grammar for terrain-optimized robot design, *ACM Trans. Graph.* 39 (6) (2019) 1–16.
- [34] A. Gupta, S. Savarese, S. Ganguli, L. Fei-Fei, Embodied intelligence via learning and evolution, *Nat. Commun.* 12 (1) (2021) 5721.
- [35] Z. Wang, B. Benes, A.H. Qureshi, C. Mousas, Evolution-based shape and behavior co-design of virtual agents, *IEEE Trans. Vis. Comput. Graphics* 30 (12) (2024) 7579–7591.
- [36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, O. Klimov, Proximal policy optimization algorithms, 2017, arXiv preprint arXiv:1707.06347.
- [37] M. Raibert, *Legged Robots that Balance*, MIT Press, 1986.
- [38] Z. Luo, Y. Dong, X. Li, R. Huang, Z. Shu, E. Xiao, P. Lu, Moral: Learning morphologically adaptive locomotion controller for quadrupedal robots on challenging terrains, *IEEE Robot. Autom. Lett.* (2024).
- [39] V. Makovychuk, L. Wawrzyniak, Y. Guo, M. Lu, K. Storey, M. Macklin, A. Handa, Isaac gym: High performance GPU based physics simulation for robot learning, in: *Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021.
- [40] N. Rudin, D. Hoeller, P. Reist, M. Hutter, Learning to walk in minutes using massively parallel deep reinforcement learning, in: *Conference on Robot Learning*, 2022, pp. 91–100.
- [41] J. Blank, K. Deb, Pymoo: Multi-objective optimization in python, *IEEE Access* (8) (2020) 89497–89509.