Deep Reinforcement Learning Based Co-Optimization of Morphology and Gait for Small-Scale Legged Robot

Ci Chen, Pingyu Xiang, Jingyu Zhang, Rong Xiong, *Senior Member, IEEE*, Yue Wang, *Member, IEEE*, and Haojian Lu, *Member, IEEE*

***Abstract*—Small-scale legged robots have found widespread utilization in various industrial and biomedical applications due to their compact size and superior locomotion capabilities. Reducing the number of actuators is often desirable to decrease the robot's size and weight, which comes at the expense of the robot's workspace. Our study proposes a method to enhance the mobility of small-scale legged robots with limited degrees of actuators (DoAs) by co-optimizing both morphology parameters and control policy. The co-optimization is formulated as a bi-level optimization problem, where the control policy is designed using deep reinforcement learning algorithms and central pattern generators (CPGs) at the lower level. The inclusion of CPGs significantly speeds up training and enables the application of simulation results in real-world scenarios. At the upper level, morphology optimization is achieved through Bayesian optimization based on dual-networks. This approach eliminates the need to train a policy for each morphology candidate from scratch, leveraging previous experience to enhance efficiency. Through simulation and physical experiments, the effectiveness of our proposed approach is demonstrated, showcasing its ability to discover optimal morphology and gait for small-scale legged robots with limited DoAs. These findings have potential long-term impacts on small-scale legged robot design and locomotion control.**

***Index Terms*—Bayesian optimization, co-optimization, deep reinforcement learning, small-scale legged robot.**

Manuscript received 7 December 2022; revised 31 August 2023; accepted 31 October 2023. Recommended by Technical Editor A. Dani and Senior Editor K. J. Kyriakopoulos. This work was supported in part by the National Key R&D Program of China under Grant 2021ZD0114500, in part by the National Natural Science Foundation of China under Grant 62373322, Grant T2293724, and Grant 62303407, in part by the Key R&D Program of Zhejiang under Grant 2022C01022 and Grant 2023C01176, and in part by the Zhejiang Provincial Natural Science Foundation of China under Grant LD22E050007. *(Corresponding authors: Haojian Lu; Yue Wang.)*

The authors are with the State Key Laboratory of Industrial Control and Technology, and Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou 310027, China (e-mail: chenci107@zju.edu.cn; xiang_py@zju.edu.cn; 12132011@zju.edu.cn; rxiong@zju.edu.cn; wangyue@iipc.zju.edu.cn; luhaojian@zju.edu.cn).

Color versions of one or more figures in this article are available at https://doi.org/10.1109/TMECH.2023.3330427.

Digital Object Identifier 10.1109/TMECH.2023.3330427

I. Introduction

DESPITE its inherent mechanical and kinematic complexity, legged locomotion is often preferred over simpler wheeled locomotion in unstructured environments. Small-scale legged robots [1], [2], [3], [4], [5], [6], with masses on the order of grams and dimensions on the order of centimeters, offer numerous advantages over their larger counterparts [7], including access to confined spaces and lower costs. However, due to size and weight constraints, these robots typically have limited degrees of actuators (DoAs) [5]. Enhancing the mobility of small-scale legged robots with limited DoAs requires further investigation into the design of mechanical structures and control policies.

A robot's capacity to interact with the environment is highly dependent on its physical design and control proficiency, which are inherently interrelated. Recently, there have been developments in approaches based on dynamics modeling and optimal control to address the robots' co-optimization problem. For instance, Ha et al. [8] showed that the design and motion parameters of robots must satisfy several constraints, which can form an implicitly defined manifold. They applied the implicit function theorem to derive relationships between the design and motion parameters. Geilinger et al. [9], [10] developed a suite of computational tools that support manual, semiautomatic, and fully automatic optimization of robots' physical dimensions. They also proposed a versatile trajectory optimization formulation that generates stable, physically valid motions for a wide range of robots utilizing legs and wheels for locomotion. In [11], a genetic optimizer selected the design and control parameters, which were then used to establish dynamic and friction models. Subsequently, trajectory optimization was performed, and the final costs serve as the optimized objective of the genetic algorithm. However, these approaches often require meticulous engineering and expert knowledge to construct motion equations and design equality/inequality constraints, which may not be consistent across different robot types.

To address the aforementioned challenges, numerous data-driven co-optimization approaches have emerged [12], [13], [14], [15], [16], [17], [18]. We provide a detailed description of these methods in the following section. These approaches learn control policies and morphology parameters by driving robots to interact with the environment through trial-and-error so that

it can be independent of the robot's dynamics. However, most of these methods focus solely on optimizing the algorithm's efficiency. Both training and validation are performed in simulators, and there is limited discussion on how to apply such methods to physical robots.

In this article, we present a data-driven approach for co-optimizing morphology parameters and control policies to enhance the mobilities of four DoAs legged robots while minimizing energy consumption. Additionally, we propose a solution to address the challenge of verifying the results obtained through simulators in a physical setting. The specific novelties and contributions of this article are as follows.

1) We develop a gait controller that integrates deep reinforcement learning (DRL) with central pattern generators (CPGs). This approach allows for the incorporation of more reliable samples, ultimately accelerating the training process. Furthermore, the trained CPG parameters have a tendency to converge to a static value, which can be directly employed in the physical system, resulting in the same effect observed in the simulation environment.
2) We develop a Bayesian optimization (BO) approach based on dual-networks to optimize morphology parameters. This method utilizes previous experiments to establish a general policy, eliminating the necessity of training a policy from scratch for each new morphology candidate during the upper-level optimization. Through this approach, training efficiency is significantly enhanced.
3) A simple four DoAs legged robot that can be easily disassembled and reassembled is constructed to verify the effectiveness of the algorithms.

The rest of this article is organized as follows. Section II provides a detailed overview of data-driven co-optimization approaches for robots. Section III describes the procedure for adjusting the morphology parameters (i.e., the scaling factor of the robot body structure size) of the robot in the simulator, as well as the construction process of the physical robot. Section IV presents our co-optimization approach. In Section V, we demonstrate the effectiveness of our proposed method through simulation environments. Section VI validates our findings in the real world. Finally, Section VII concludes this article.

II. Related Works

Data-driven co-optimization of morphology and policy approaches can be categorized into two groups: 1) robot topological configuration changing and 2) unchanging.

Regarding the first category, Hejna et al. [12] assumed that the optimal robot morphology structure was independent of actions and represented the robot structure using a graph. They used graph neural networks (GNNs) to learn fitness under different actions and morphologies during the structure optimization process. The information-theoretic objective was then used to rank the agent, and the genetic algorithm (GA) was applied to optimize the morphology. Wang et al. [13] proposed neural graph evolution, which performed an evolutionary search in graph space by iteratively evolving graph structures using simple mutation primitives. Key to this approach is parameterizing control policies with GNNs, allowing for skill transfer from previously evaluated designs during the graph search. Gupta et al. [14] introduced deep evolutionary reinforcement learning, a novel computational framework that could evolve diverse agent morphologies to learn challenging locomotion and manipulation tasks in a complex environment using only low-level egocentric sensory information. Through this framework, the fitness of an agent can be rapidly transferred within a few generations of evolution from its phenotypic ability to its genotypically encoded morphology through the Baldwin effect. Although these methods have achieved certain effectiveness in simulation environments, they are difficult to apply to physical robots because the optimized structures are somewhat peculiar and the arrangement of the motors is also unreasonable. The nonstandard topology makes it challenging to install the motors at predetermined angles.

For the second category, in [15], the body morphology design was parameterized by a small set of learnable parameters, which were set once at the beginning of a rollout. The control parameters and morphology parameters were optimized simultaneously. Hu et al. [16] proposed using low fidelity levels of evaluation to optimize robot morphology, wherein easier tasks were used to test candidate robot designs instead of evaluating their performance on difficult tasks. Luck et al. [17] suggested using a learned morphology-conditioned state-action value function as the surrogate objective to estimate candidate design performance and guide design optimization, thereby avoiding the execution of a larger number of episodes. In [18], a distribution over designs was maintained, and reinforcement learning was used to optimize a control policy to maximize the expected reward over the design distribution. Most of the aforementioned methods adopt robots proposed by Gym [19], such as HalfCheetah and Ant, which exhibit a significant gap between simulated and real robots. However, these methods are more promising for deployment in real environments than topologically changing methods, and the proposed method can also be classified into this category.

III. System and Modeling

A. Adjustment of Morphology Parameters

In our simulation experiments, we represent the morphology configuration of the robot using the unified robot description format (URDF) file. Due to our algorithm's objective of modifying the robot's morphology parameters, it is not feasible to load a mesh file to create a morphology structure similar to that of an ingenious robot. Therefore, we utilize cubes and cylinders as representations in our study. In this article, we specifically focus on a four DoAs legged robot, consisting of four legs, each with one DoA. To maintain symmetry, we ensure that the length and radius of the left leg and right leg remain consistent throughout. Consequently, the design of the robot's morphology parameters centers around a set of scaling factors, including body width ξ_1, body length ξ_2, front leg length ξ_3, front leg radius ξ_4, hind leg length ξ_5, and hind leg radius ξ_6.

When modifying these parameters, the URDF file of the robot must be updated accordingly. The values to be changed for the robot body include the mass and inertia parameters of the inertial attribute, as well as the geometry parameters of the collision and visual attributes. We use the following calculation formula to determine these parameters:

$$\mathbf{I_{body}} = \begin{bmatrix} I^{xx}_{\text{body}} \\ I^{yy}_{\text{body}} \\ I^{zz}_{\text{body}} \end{bmatrix} = \frac{m_{\text{body}}}{12} \begin{bmatrix} h^2_{\text{body}} + w^2_{\text{body}} \\ h^2_{\text{body}} + l^2_{\text{body}} \\ l^2_{\text{body}} + w^2_{\text{body}} \end{bmatrix} \tag{1}$$

$$s.t. \quad m_{\text{body}} = \rho_{\text{body}} \times h_{\text{body}} \times w_{\text{body}} \times l_{\text{body}} \tag{2}$$

$$w_{\text{body}} := w_{\text{body}} \times \xi_1 \tag{3}$$

$$l_{\text{body}} := l_{\text{body}} \times \xi_2 \tag{4}$$

where w_{body}, l_{body}, and h_{body} denote the width, length, and height of the body, respectively. ρ_{body} and m_{body} represent the density and mass of the body, respectively. I^{xx}_{body}, I^{yy}_{body}, and I^{zz}_{body} are defined as the moment of inertia of the body.

The parameters of the robot's leg, including length, radius, mass, moment of inertia, and origin, must be adjusted accordingly. The calculation formula is provided as follows:

$$\mathbf{I_{leg}} = \begin{bmatrix} I^{xx}_{\text{leg}} \\ I^{yy}_{\text{leg}} \\ I^{zz}_{\text{leg}} \end{bmatrix} = m_{\text{leg}} \begin{bmatrix} \frac{1}{12}(3 \times r^2_{\text{leg}} + l^2_{\text{leg}}) \\ \frac{1}{12}(3 \times r^2_{\text{leg}} + l^2_{\text{leg}}) \\ \frac{1}{2} \times r^2_{\text{leg}} \end{bmatrix} \tag{5}$$

$$s.t. \quad m_{\text{leg}} = \rho_{\text{leg}} \times \pi \times r^2_{\text{leg}} \times l_{\text{leg}} \tag{6}$$

$$l_{\text{leg}} := l_{\text{leg}} \times \xi_{i(i=3,5)} \tag{7}$$

$$r_{\text{leg}} := r_{\text{leg}} \times \xi_{i(i=4,6)} \tag{8}$$

$$z_{\text{leg}} = l_{\text{leg}} \times \xi_{i(i=3,5)}/2 \tag{9}$$

where l_{leg}, r_{leg}, and z_{leg} are the length, radius, and the z-axis position value of origin of legs, respectively. Besides, ρ_{leg} and m_{leg} represent the density and mass of the leg, respectively. In addition, I^{xx}_{leg}, I^{yy}_{leg}, and I^{zz}_{leg} denote the moment of inertia of legs.

Furthermore, as the length and width of the body link have been altered, adjustments must be made to the coordinate origins of the joints connecting the leg links and body link. It is worth noting that the URDF file follows a tree structure, where the coordinate origin of the joints is relative to the body link. Thus, the revised value should be calculated using the following formula:

$$x^j_{\text{joint}} = \begin{cases} (-1)^{j+1} \times \left[l_{\text{body}} \times \frac{\xi_2}{2} - \varepsilon_{\text{edge}} \right]_{(j=1,4)} \\ (-1)^{j} \times \left[l_{\text{body}} \times \frac{\xi_2}{2} - \varepsilon_{\text{edge}} \right]_{(j=2,3)} \end{cases}$$

$$y^j_{\text{joint}} = (-1)^{j+1} \times \left[w_{\text{body}} \times \frac{\xi_1}{2} - \varepsilon_{\text{edge}} \right]_{(j=1,2,3,4)} \tag{10}$$

where $j = 1$ corresponds to the front left (FL) leg joint, $j = 2$ denotes the front right (FR) leg joint, $j = 3$ signifies the hind left (HL) leg joint, and $j = 4$ represents the hind right (HR) leg joint. The parameter $\varepsilon_{\text{edge}}$ refers to the distance between the joints and the body link's edge. Fig. 1 shows a schematic diagram of the robot with scale factors.

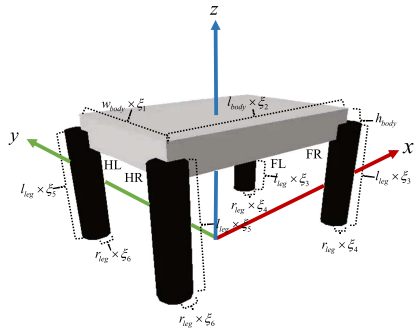

Fig. 1. Description of the morphology parameters of the four DoAs legged robot.

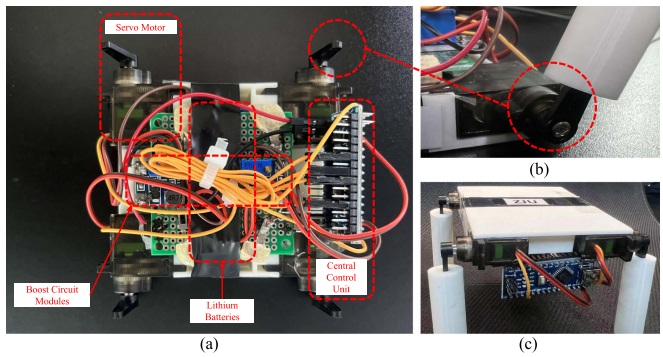

Fig. 2. Four DoAs legged robot. (a) The servo motors, boost circuit modules, lithium batteries, and central control unit are integrated into the back of the robot body. (b) Leg design that can be easily detached. (c) The overall display of the robot.

Following the modification of the parameters, the *xmltodict* library is utilized to overwrite the modified parameters into their corresponding parameters in the original URDF file. Subsequently, the resulting new URDF file accurately reflects the new robot morphology configuration. It is worth noting that while we chose a four DoAs legged robot as an example, our method is applicable to robots with various topologies.

B. Introduction of Four DoAs Legged Robot

We have designed a legged robot with four DoAs based on the simulation model, as shown in Fig. 2. The robot consists of two main components: the body and the legs, both of which have been 3-D-printed using polylactic acid (PLA) material. Following the design from the simulation, the robot's body has been shaped as a cube, while the legs are cylindrical. At the back of the robot's body, there is a central control unit equipped with an ATmega328 chip, responsible for governing the movement of the robot. To connect the legs to the body, four servo motors (MG90S) are used, providing torque ranging from 2.0 kg/cm to 2.8 kg/cm. For the power supply, we employ two rechargeable 3.7 V cylindrical lithium batteries (LR14500) as the robot's power module. To adjust the voltage to 5 V for the servo motors and 6-9 V for the main control chip, we incorporate two adjustable boost circuit modules (SX 1308 dc-dc) into the power module. We have integrated the control system, four servo motors, and a power module into the robot's body, giving it a cuboid appearance.

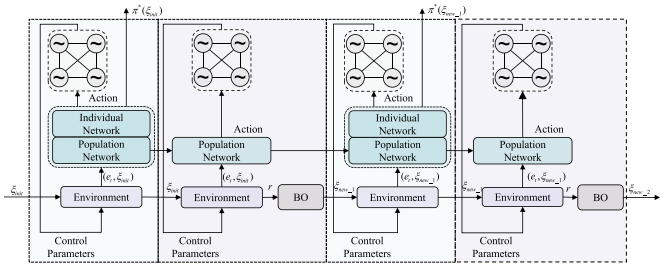

Fig. 3. Proposed method is composed of two optimization procedures. The lower-level optimization, marked in cyan, generates the optimal control policies for the specific morphology parameters. Conversely, the upper-level optimization, marked in purple, produces optimized morphology parameters. These two optimization procedures are executed alternately.

Additionally, we have designed detachable and assemblable legs to facilitate morphology optimization experiments.

IV. Methodology

A. Central Pattern Generator

CPGs are neural circuits prevalent in most vertebrates, which generate coordinated patterns of rhythmic movements, such as fishes swimming [20], [21], [22] and snakes moving in a serpentine manner [23]. The CPGs are a popular option for designing gaits for robot locomotion due to their ability to replicate different gaits merely by manipulating the relative coupling phase biases between oscillators. These CPGs can be modeled as a network of coupled nonlinear oscillators whose dynamics are determined by a set of differential equations.

$$\dot{\phi}_i = 2\pi f_i + \sum_{j\in\Omega_i} \mu_{ij}(\phi_j - \phi_i - \varphi_{ij}) \tag{11}$$

$$\ddot{r}_i = a_r^2(R_i - r_i) - 2a_r\dot{r}_i \tag{12}$$

$$\ddot{x}_i = a_x^2(X_i - x_i) - 2a_x\dot{x}_i \tag{13}$$

$$\theta_i = x_i + r_i\cos(\phi_i) \tag{14}$$

where ϕ_i, r_i, and x_i are three state variables, which represent the phase, amplitude, and offset of each oscillator, while θ_i is the output of oscillator i, representing the desired deflection angle of the corresponding joint. Control parameters for the desired frequency, amplitude, and offset of each oscillator are f_i, R_i, and X_i, respectively, and φ_{ij} is the desired phase bias between oscillator i and j. The coupling weights μ_{ij} determine how the oscillators influence each other, and constant positive gains, a_r and a_x, control how quickly the amplitude and offset variables can be modulated. Ω_i denotes the set of all oscillators that can have a coupling effect on oscillator i. The subscripts $i = 1, 2, 3$, and 4 denote the FL leg, FR leg, HL leg, and HR leg of the legged robot, respectively.

Such a CPG model can spontaneously generate rhythmic output signals to make the four DoAs legged robot move and easily change its locomotion behavior by adjusting the input parameters. As the movement speed of the robot will increase with the increase of the frequency f_i and the amplitude R_i [24], if they are not fixed, the algorithm will tend to maximize the value of them within the feasible range. Therefore, we specify the values of f_i and R_i and assign them reasonable values. For the positive gains a_r and a_x, the values are taken from [24]. In summary, the parameters are set as follows: $f_{i,i=1,2,3,4} = 10$ Hz, $R_{i,i=1,2,3,4} = 0.4$ rad, $X_{i,i=1,2,3,4} = 0.04$ rad, $a_r = 20$, $a_x = 20$, $\mu_{ii} = 0$ ($i.e.$, no self-couplings), and $\mu_{ij} = 20$ for $i \neq j$. The resulting CPG model only has φ_{ij} as an undetermined parameter represented by a 4×4 matrix, which can be calculated using the following formula

$$\varphi_{ij} = \varphi_j - \varphi_i. \tag{15}$$

So the values that we need to learn are four φ_i (for $i = 1$, 2, 3, 4).

B. Optimization Framework

The proposed method's framework is depicted in Fig. 3. To optimize the legged robot's morphology parameters and gait parameters, a bilevel optimization model is employed.

$$\max_{\xi\in\Xi} \mathcal{F}(\pi^*(\xi), \xi) \tag{16}$$

$$\text{s.t. } \pi^*(\xi) = \arg\max_{\pi\in\Pi} \mathcal{J}(\pi, \xi) \tag{17}$$

where π represents the control policy determining the robot's gait, ξ denotes the morphology parameters, and $\mathcal{F}(\cdot)$ and $\mathcal{J}(\cdot)$ are objective functions for the upper and lower optimization layers, respectively. With predefined morphology parameters, we first perform the lower-level optimization to obtain the optimized gaits and then perform the upper-level optimization to obtain the optimized morphology parameters.

1) Lower-Level Optimization: Legged robot locomotion can be viewed as Markov decision processes (MDPs)-$\langle S, A, R, p, \gamma\rangle$, comprising state space S, action space A, scalar reward function R, transition dynamics p, and discount factor γ. At each time step t, the state of the agent and the environment can be jointly described by a state vector $s \in S$. To incorporate morphology parameters into the optimization process, ξ is encoded into s, forming the following expression:

$$s_t^\xi = CONCAT(e_t, \xi) \tag{18}$$

where e_t and ξ denote observations from the environment at time step t and morphology parameters, respectively. $CONCAT(\cdot)$ means to splice all elements. The dimension of the former is about six to seven times that of the latter, which is not conducive to subsequent training. To address this issue, we incorporate a fully connected layer into the network to balance the dimensions, such that the former's dimension is twice that of the latter, as illustrated in Fig. 4.

The action $a_t \in A$ is the phase difference of oscillators, which is used to determine the gait, as described in (15). The CPG executes a cycle whenever an action is selected, and the resulting state s_{t+1}^ξ is attained through a transition $p(s_{t+1}^\xi | s_t^\xi, a_t)$. The reward is used to evaluate the chosen action which is denoted by $r(s_t^\xi, a_t)$. The action is determined by a policy $\pi \in \Pi$. It maps states into actions. Given the morphology parameters, the lower-level optimization aims to optimize the policy/gaits of

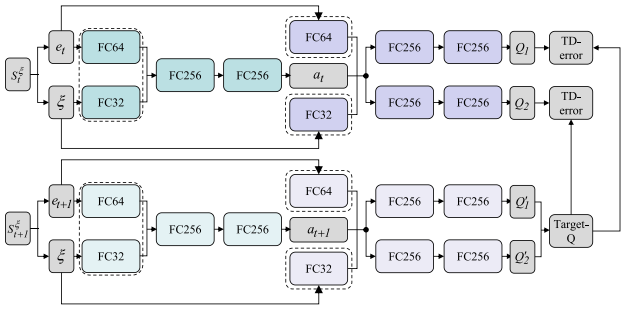

Fig. 4. Architecture of population network/individual network. The cyan parts represent the actor network, the purple parts represent the critic network, and the gray parts represent input or output data. The number of fully connected (FC) layers indicates the output dimensions. Additionally, the light cyan parts represent the target network for the actor, while the light purple parts represent the target network for the critic.

robots to maximize the expectation of the sum of discounted rewards. Its evaluation function is as follows:

$$\mathcal{J}(\pi,\xi)=\mathbb{E}\left[\sum_{t=0}^{H}\gamma^t r\left(s_t^\xi,a_t\right)\middle| a_t\sim\pi\left(\cdot|s_t^\xi\right)\right] \tag{19}$$

where H refers to the horizon. The objective of the lower level is to select an optimal policy π^* that maximizes $\mathcal{J}(\pi,\xi)$.

$$\pi^*(\xi)=\arg\max_{\pi\in\Pi}\mathcal{J}(\pi,\xi). \tag{20}$$

We utilize Soft Actor-Critic (SAC) [25], [26] to solve the lower-level optimization problem. To ensure consistency with the optimization objective of SAC, we augment the objective function with the entropy term $\mathcal{H}(\cdot)$. This modification encourages agents to explore and promotes a better exploration-exploitation trade-off. As a result, (19) is redefined as follows:

$$\begin{aligned}&\mathcal{J}(\pi,\xi)\\&=\mathbb{E}\left[\sum_{t=0}^{H}\gamma^t r\left(s_t^\xi,a_t\right)+\alpha\mathcal{H}\left(\pi\left(\cdot|s_t^\xi\right)\right)\middle| a_t\sim\pi\left(\cdot|s_t^\xi\right)\right]\end{aligned} \tag{21}$$

where α represents the weight that balances between the cumulative rewards and entropy. Since (11)–(13) are represented by differential equations, discretization of the CPG model is necessary to facilitate its use on a computer. Using the Euler difference method, (11)–(14) can be discretized into

$$\begin{cases}\dot r_i(k+1)=\dot r_i(k)+T[a_r^2(R_i(k)-r_i(k))-2a_r\dot r_i(k)]\\ r_i(k+1)=r_i(k)+T\dot r_i(k+1)\\ \dot x_i(k+1)=\dot x_i(k)+T[a_x^2(X_i(k)-x_i(k))-2a_x\dot x_i(k)]\\ x_i(k+1)=x_i(k)+T\dot x_i(k+1)\\ \phi_i(k+1)=\phi_i(k)+T(2\pi f_i+\sum_{j\in\Omega_i}\mu_{ij}(\phi_j-\phi_i-\varphi_{ij}))\\ \theta_i(k)=x_i(k)+r_i(k)\cos(\phi_i(k))\end{cases} \tag{22}$$

where k is the number of iterations, and T is the update interval. The update steps in one cycle are as follows:

$$N=\frac{1}{f\times T}. \tag{23}$$

Given that the CPG is periodic, it presents challenges when combined with conventional DRL stepping training methods. To address this issue, we propose a novel periodic training method based on the CPG. In the interaction between the legged robot and its environment, the CPG is initially provided with a set of parameters by the policy network π. CPG then updates according to (22), and the resulting updated values are used as position control commands for the robot's leg joints. To differentiate between these steps, the step of DRL is referred to as simply "step," while the step of CPG update is called the "mini-step". After N mini-steps (one cycle), a step of DRL is completed. At each mini-step, the environment returns states and rewards that require adjustment. We adjust the states and rewards using the following formula:

$$\begin{cases}s_t^\xi=\frac{1}{N}\sum_{n=tN}^{(t+1)N}s_n^\xi\\ r_t=\frac{1}{N}\sum_{n=tN}^{(t+1)N}r_n\\ s_{t+1}^\xi=\frac{1}{N}\sum_{n=tN}^{(t+1)N}s_{n+1}^\xi\end{cases} \tag{24}$$

where subscript n denotes the number of mini-steps, while subscript t denotes the number of steps. In this way, we combine CPG with DRL for better gait training. During the training process, the tuple $\left\langle s_t^\xi,a_t,r_t,s_{t+1}^\xi\right\rangle$ is stored in the replay buffers.

2) Upper-level Optimization: To tackle the upper-level optimization problem, we utilize the BO method [27], which mimics the optimization technique employed by humans to optimize an unknown function. The goal of morphology optimization is to select morphology parameters that can maximize the expectation of the accumulative rewards given the optimal policy, represented by

$$\mathcal{F}(\pi^*(\xi),\xi)=\mathbb{E}\left[\sum_{t=0}^{H}\gamma^t r\left(s_t^\xi,a_t\right)\middle| a_t\sim\pi^*\left(\cdot|s_t^\xi\right)\right]. \tag{25}$$

Note that although (25) and (19) are very similar, their optimization goals differ. The goal of (19) is to provide a predefined morphology parameter and optimize the gait, while the goal of (25) is to provide an optimal gait and optimize the morphology parameters.

Designing an optimal control policy for each morphology candidate is a time-consuming process. To facilitate the upper-level optimization, we have devised dual-networks consisting of the following two identical networks: 1) the individual network (including Q_{ind} and π_{ind}) and 2) the population network (including Q_{pop} and π_{pop}). The former is trained using experiences obtained from the current morphology parameters, enabling it to learn a policy that is better suited for the specific morphology parameters. Meanwhile, the latter is trained using experiences gathered by individual networks with different morphology parameters, allowing it to generalize effectively across various morphology parameters. When encountering a new morphology candidate, the population network is expected to provide a general policy that can be utilized for fitness computation, eliminating the need to train a policy for each new morphology candidate from scratch. This approach significantly enhances the algorithm's efficiency. The formulas for updating the parameters of each

network are as follows:

$$\mathcal{L}_{Q_{\text{pop}}}(\tau_i) = \mathbb{E}_{(s_t^\xi, a_t)\sim D_{\text{pop}}}\Big[\frac{1}{2}(r(s_t^\xi, a_t) + \gamma\Big(\min_{i=1,2} Q_{\bar{\tau}_i}\left(s_{t+1}^\xi, a_{t+1}\right) - \alpha \log \pi_\omega\left(a_{t+1}|s_{t+1}^\xi\right)\Big) - Q_{\tau_i}(s_t^\xi, a_t))^2\Big] \tag{26}$$

$$\mathcal{L}_{\pi_{\text{pop}}}(\omega) = \mathbb{E}_{s_t^\xi\sim D_{\text{pop}},\varepsilon\sim\mathcal{N}(0,1)}\Big[\alpha \log \pi_\omega\left(f_w\left(s_t^\xi;\varepsilon\right)|s_t^\xi\right) - \min_{i=1,2} Q_{\tau_i}\left(s_t^\xi, f_w\left(s_t^\xi;\varepsilon\right)\right)\Big] \tag{27}$$

$$\mathcal{L}_{Q_{\text{ind}}}(\tau'_i) = \mathbb{E}_{(s_t^\xi, a_t)\sim D_{\text{ind}}}\Big[\frac{1}{2}(r\left(s_t^\xi, a_t\right) + \gamma\Big(\min_{i=1,2} Q_{\bar{\tau}'_i}(s_{t+1}^\xi, a_{t+1}) - \alpha \log \pi_{\omega'}\left(a_{t+1}|s_{t+1}^\xi\right)\Big) - Q_{\tau'_i}\left(s_t^\xi, a_t\right))^2\Big] \tag{28}$$

$$\mathcal{L}_{\pi_{\text{ind}}}(\omega') = \mathbb{E}_{s_t^\xi\sim D_{\text{ind}},\varepsilon\sim\mathcal{N}(0,1)}\Big[\alpha \log \pi_{\omega'}\left(f_{\omega'}\left(s_t^\xi;\varepsilon\right)|s_t^\xi\right) - \min_{i=1,2} Q_{\tau'_i}\left(s_t^\xi, f_{\omega'}\left(s_t^\xi;\varepsilon\right)\right)\Big] \tag{29}$$

where the target network for the critic is denoted as $\bar{\tau}$, while the subscript $i = 1, 2$ signifies the number of Q-networks. The network parameters of Q_{pop} and π_{pop} are represented by τ and ω, respectively, while the network parameters of Q_{ind} and π_{ind} are represented by τ' and ω', respectively. Random Gaussian noise is denoted by ε, and the function f_ω represents reparameterization. In this way, the upper-level optimization objective can be simplified to

$$\mathcal{F}(\pi^*(\xi), \xi) \approx \mathcal{F}(\pi_{\text{pop}}(\xi), \xi) = \left[\sum_{t=0}^{H} \gamma^t r\left(s_t^\xi, a_t\right) \middle| a_t \sim \pi_{\text{pop}}\left(\cdot|s_t^\xi\right)\right]. \tag{30}$$

The objective of BO is to find ξ that maximizes the value of the objective function. In each iteration, BO learns a model $\mathcal{M} : \xi \mapsto \mathcal{F}(\pi_{\text{pop}}, \xi)$ from the dataset of previously evaluated parameters and corresponding objective values stored in $D_{BO} = \{\xi, \mathcal{F}(\pi_{\text{pop}}, \xi)\}$, where $\mathcal{F}(\pi_{\text{pop}}, \xi)$ is obtained using (30). A commonly used model in BO for learning the underlying objective is the Gaussian process (GP), which is also considered in this study. Subsequently, the learned GP model $\mathcal{M}$ is utilized to perform optimization by employing an acquisition function $\psi_i(\xi)$. This acquisition function determines the balance between exploration and exploitation. Specifically, we utilize the Gaussian process upper confidence bound (GP-UCB) as the acquisition function, defined as follows:

$$\xi_i = \arg\max_{\xi\in\Xi} \psi_i(\xi) = \arg\max_{\xi\in\Xi} \mu_{i-1}(\xi) + \beta^{1/2}\sigma_{i-1}(\xi) \tag{31}$$

Algorithm 1: Proposed Algorithm.

1: Initialize replay buffers: D_{pop}, D_{ind};
2: **for** each iteration **do**
3: $\pi_{ind} \leftarrow \pi_{pop}$, $Q_{ind} \leftarrow Q_{pop}$;
4: Initialize and empty D_{ind};
5: $\xi = \xi_{new}$;
6: **for** steps t in episode length **do**
7: Wrap ξ according to (18);
8: Use π_{ind} to select action: $a_t \sim \pi_{ind}(a_t|s_t^\xi)$;
9: **for** mini-step n in N **do**
10: Update CPG parameters according to (22);
11: Execute θ to joints;
12: Observe states s_{n+1}^ξ and receive rewards r_n;
13: **end for**
14: Adjust states and rewards according to (24);
15: Store transition $\left\langle s_t^\xi, a_t, r_t, s_{t+1}^\xi\right\rangle$ in replay buffers;
16: Train Q_{pop} and π_{pop} by (26) and (27);
17: Train Q_{ind} and π_{ind} by (28) and (29);
18: **end for**
19: **for** step i in BO update numbers **do**
20: Find ξ_i according to (31);
21: Wrap ξ_i according to (18);
22: Use π_{pop} to select action: $a_t \sim \pi_{pop}(a_t|s_t^\xi)$;
23: Update CPG parameters and execute θ to get rewards as in lines 9-13;
24: Calculate $\mathcal{F}(\pi_{pop}, \xi_i)$ according to (30);
25: Augment $D_{BO}^{1:i} = \{D_{BO}^{1:i-1}, (\xi_i, \mathcal{F}(\pi_{pop}, \xi_i))\}$ and update the GP;
26: **end for**
27: $\xi_{new} = \arg\max_i \mathcal{F}(\pi_{pop}, \xi_i)$
28: **end for**

where $\mu_{i-1}(\xi)$ and $\sigma_{i-1}^2(\xi)$ represent the mean and variance of $\mathcal{M}$, respectively. The hyperparameter β controls the balance between exploration and exploitation. Once ξ_i is obtained, it is evaluated using (30), and the corresponding measurement $\mathcal{F}(\pi_{\text{pop}}, \xi_i)$ is added to the dataset before starting a new iteration. When the BO procedure reaches the predetermined steps, it returns the morphology parameters corresponding to the highest objective function value. These parameters are then used as the new morphology parameter ξ_{new} for a new round of lower-level optimization. The pseudocode of our co-optimization method is presented in Algorithm 1.

V. Simulation Results and Analysis

In this section, we aim to validate the effectiveness of our proposed approach in a simulation environment. To begin with, we compare the efficacy of the morphology optimization method against two baseline approaches. Subsequently, we analyze the outcomes obtained from the morphology optimization method. Furthermore, we conduct comparative experiments between our proposed gait optimization method and Vanilla DRL, particle swarm optimization (PSO), and classical gaits.

TABLE I
HYPERPARAMETERS SETTING

	Hyperparameters	Value
Lower-Optimization Setting	Discount factor	0.99
	Reward scale	1
	Target update weight	5×10^{-3}
	Policy update frequency	1
	Actor learning rate	3×10^{-4}
	Critic learning rate	3×10^{-4}
	Batch size	256
	Non linearly	ReLU
	Optimizer	Adam
	Horizon	1000
	Capacity of D_{ind}	1×10^{6}
	Capacity of D_{pop}	1×10^{7}
Network Architecture	Population network	As shown in Fig.4
	Individual network	As shown in Fig.4
Upper-Optimization Setting	Bayesian optimization steps	30
	Random exploration steps	30

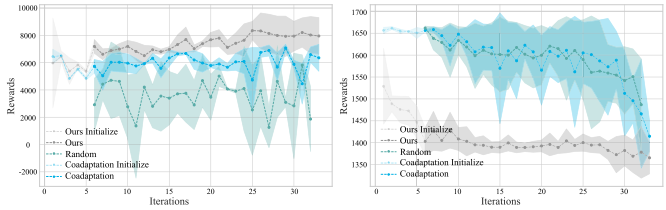

Fig. 5. Comparison of the proposed method with two baselines. (a) Comprehensive performance comparison. (b) Energy costs comparison. The x-axis represents the number of optimization iterations in the upper level, while the y-axis denotes the reward achieved under this morphology parameter. The results presented are average across five experiments, and the shaded areas represent the variance of these five experiments. It is worth that for the first five rounds, we manually specified the morphology parameters to initialize the dual-networks, and the respective optimization methods start from the sixth round.

A. Setup

The proposed model and comparative experiment are conducted in a simulation environment created by PyBullet. The training is performed on an NVIDIA GeForce GTX 2080Ti GPU, and networks are established by PyTorch. Prior to morphology optimization, we manually designed five groups of morphology parameters to initialize the dual-networks. The details of the hyperparameters can be found in Table I.

The state space is 40-D, which includes body position, body orientation, body linear velocity, body angular velocity, joint position, joint position history, joint velocity, joint velocity history, and previous actions. The action space has four dimensions, representing the phase difference of the CPGs. The objective of the four-legged robot is to move forward as quickly as possible while minimizing the energy cost. Therefore, the reward function is defined as $r_t = \frac{x_{t+1}-x_t}{\alpha_1} - \alpha_2|\theta|^2 + r_{\text{alive}}$. Here, x_t represents the position along the x-axis at time step t, θ denotes the robot joint angle, α_1 and α_2 are weights for forward rewards and energy costs, respectively, and r_{alive} denotes the survival bonus. These parameters are manually formulated. The robot model is represented by a URDF file, and the morphology parameters that we modified are identical to those described in Section III-A.

B. Morphology Optimization

1) Comparison With Baselines: In this section, we evaluate and compare the proposed method with two baselines. It is worth noting that for both baselines, we replace the policy optimization component with the gait optimization method proposed in our approach to enable a more accurate comparison of the morphology optimization aspect. To begin with, we introduce the two baseline algorithms.

Uniform Sampling: This baseline randomly samples morphology parameters uniformly within the parameter range, which serves as the lower bound for optimization.

Coadaptation: This method employs a learned morphology conditioned state-action value function as the surrogate objective for the PSO method to select the morphology parameters.

Fig. 5(a) illustrates the comprehensive performance comparison of the proposed method, uniform sampling, and coadaptation. The reward values, in descending order, are as follows: the proposed method, coadaptation, and uniform sampling. The results of uniform sampling are significantly uneven and do not exhibit an upward trend with optimization progress. In contrast, the proposed method starts with a higher reward than the manually set initialization value (represented by the light gray line), and the rewards consistently increase as the optimization progresses, demonstrating the superiority of our approach. The coadaptation method shows significant fluctuations in the later stages of optimization. We suspect that this may be due to the introduction of randomly initialized environment state parameters during the training process of the state-action value function. These parameters have a larger dimension compared to the morphology parameters, which can result in the trained state-action value function failing to accurately capture the expected cumulative values, leading to an inaccurate objective function for morphology optimization. Furthermore, we conduct an independent T-test to compare the proposed method with the two baselines. The p-values for the proposed method versus random sampling and the proposed method versus coadaptation were 4.5279×10^{-19} and 2.8624×10^{-14}, respectively. Both p-values are less than the threshold of 0.05, indicating that the mean value of the proposed method is statistically significantly higher than that of the two baselines.

Fig. 5(b) illustrates the evolution of energy costs throughout the optimization process. As depicted in the figure, our algorithm demonstrates a tendency to reach a relatively stable energy loss value during the early stages of optimization. While the two comparison methods also reduce energy loss, they do so at a slower rate compared to the proposed method. Furthermore, the proposed method exhibits relatively small variance, indicating its robustness to different random seeds.

2) Result Analysis and Verification: To validate the effectiveness of our morphology optimization results, we select two morphology parameters, namely front leg length scale factor ξ_3 and hind leg length scale factor ξ_5, and plot them on the x-axis and y-axis in Fig. 6, respectively. The figure clearly shows that

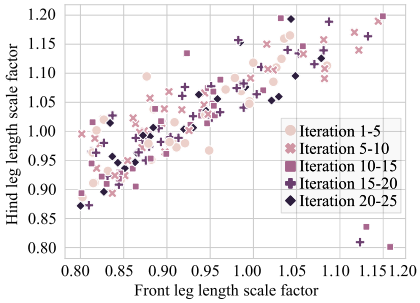

Fig. 6. Morphology optimization result, the x-axis is the front leg length scale factor, and the y-axis is the hind leg length scale factor. The shape of points represents the iteration steps of morphology optimization, with five steps as a batch.

regardless of the order of optimization iteration, most of the optimization results are located on the upper left side of the figure. This finding indicates that longer hind legs tend to result in faster running speeds for four DoAs legged robots across various gaits. It is worth noting that three points lie outside the range, which may have resulted from the BO method getting stuck in a local minimum during the upper layer optimization. However, since these outliers account for a very small proportion of the hundreds of optimization results obtained, we have ignored them. To confirm this observation, we keep the other morphology parameters fixed and discretize the values within the range of front and hind leg length scale factors. We then design various combinations and test the distances traveled by the robots under different leg length ratios and various gaits (we use three classical gaits, which are detailed in the next section). The results are recorded in Table II.

As shown in Table II, the average distances (marked with bold entities) traveled by the robot under three gaits are greater when the hind legs are longer than the front legs ($\xi_3 < \xi_5$) compared to the corresponding distances when the hind legs are shorter than the front legs ($\xi_3 > \xi_5$) and when both leg lengths are the same ($\xi_3 = \xi_5$). These findings support the earlier observation that longer hind legs tend to result in faster running speeds for four DoAs legged robots across most gaits. Asymmetrical body structures are also relatively common in nature. [28] points out that quadrupeds have acquired an asymmetrical body shape along the fore-aft direction through evolution, which has helped them achieve adaptive and efficient locomotion. Besides, [29] states that asymmetrical body structures are conducive to high-speed movement. Asymmetric designs are also utilized in the design of small-legged robots, such as [30].

Furthermore, Table II suggests that under different leg length ratios, the distances traveled by the trot gait are generally greater than those of the walk gait. However, when $\xi_5 - \xi_3 \geq 0.2$ (marked with underlines in Table II), it can be observed that the distance covered by the trot gait is less than that of the walk gait. This finding indicates that the diagonal trot gait is not suitable when the hind legs are much longer than the front legs. This phenomenon is intriguing since it suggests that the optimal gait for the four DoAs legged robot may vary depending on its morphology parameters.

TABLE II
DISTANCES UNDER DIFFERENT GAITS AND LEG LENGTHS

Leg length ξ_3	Leg length ξ_5	Walk	Trot	Pace
			$\xi_3 < \xi_5$	
0.8	0.9	98.62	238.88	29.21
0.8	1.0	241.87	264.96	17.04
0.8	1.1	285.56	229.18	72.50
0.8	1.2	358.70	153.52	77.68
0.9	1.0	163.92	268.06	24.91
0.9	1.1	287.77	284.34	33.80
0.9	1.2	334.27	233.11	97.39
1.0	1.1	228.41	336.94	32.60
1.0	1.2	324.71	289.37	48.00
1.1	1.2	289.77	321.22	39.62
The mean values		**261.36**	**261.96**	**47.28**
ξ_3	ξ_5		$\xi_3 > \xi_5$	
0.9	0.8	51.66	164.75	16.52
1.0	0.8	89.48	121.03	27.82
1.0	0.9	38.82	143.46	10.56
1.1	0.8	124.01	323.79	35.37
1.1	0.9	123.53	159.86	32.57
1.1	1.0	26.78	130.60	6.67
1.2	0.8	70.34	216.77	87.23
1.2	0.9	134.57	353.01	47.70
1.2	1.0	141.94	190.49	41.34
1.2	1.1	36.66	122.59	10.01
The mean values		**83.78**	**192.64**	**31.58**
ξ_3	ξ_5		$\xi_3 = \xi_5$	
0.8	0.8	34.20	68.35	2.31
0.9	0.9	21.70	42.53	3.05
1.0	1.0	26.11	60.47	3.91
1.1	1.1	31.32	62.92	6.67
1.2	1.2	36.66	69.52	9.30
The mean values		**30.00**	**60.76**	**5.05**

C. Gait Optimization

In this section, we first conduct an ablation experiment to evaluate the gait learned by DRL without the CPG (vanilla DRL). Next, we compare the gaits optimized using our proposed method with those obtained through PSO. Finally, we introduce three classical gaits of quadruped robots and compare the trained gait with the classical gaits under the same morphology. The following sections provide detailed information about our methods and conclusions.

1) Compared With Vanilla RL: To compare the CPG-based DRL gait optimization method with the vanilla DRL method (SAC) and demonstrate the superiority of the former, a comparative experiment is conducted. The proposed method includes a cycle of the CPG update steps (referred to ministeps in Section IV-B1) within each DRL step. Thus, the instruction sent to the robot in one episode equals the number of DRL steps in one episode multiplied by the number of CPG update steps for one cycle. To ensure fairness, the number of steps in each episode of the vanilla DRL method is set to the number of steps actually sent to the robot. Fig. 7 presents the comparison of reward curves between the proposed method and the vanilla DRL for two manually specified morphology parameters. The proposed method achieves significantly higher reward values than the vanilla DRL method and converges more easily. Additionally, its variance is relatively small, indicating that it is less sensitive to random seeds. We speculate that this improvement is due to the fact that the rhythm output of the CPG does not produce strange gaits, which could cause robot instability, resulting in

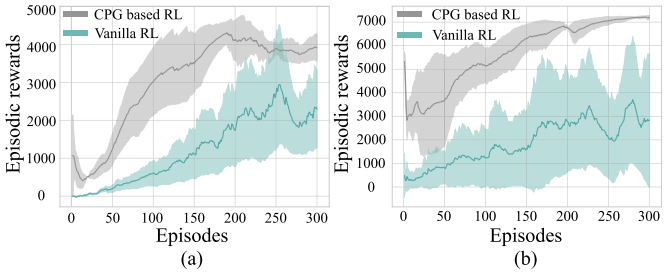

Fig. 7. Comparison of the reward curves of CPG-based DRL and vanilla DRL method. The x-axis denotes the episode numbers, while the y-axis represents the accumulative rewards in one episode for the proposed method and vanilla RL method. The results are the average of five experiments conducted with different random seeds. The solid lines depict the mean of the five experiments, and the shaded areas represent the variance of the results. (a) and (b) display the cumulative return values for different manually setting morphology parameters, respectively.

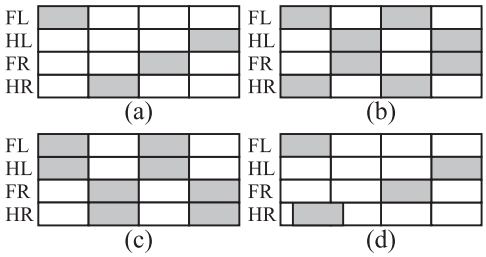

Fig. 8. Comparison of classical gaits and trained gait. (a) Walk gait phase diagram. (b) Trot gait phase diagram. (c) Pace gait phase diagram. (d) Trained gait phase diagram. The motion of legged robots consists of two phases: stance and swing. During the stance phase, the legs are in contact with the ground to support and propel the body forward, while during the swing phase, the legs are lifted and swinging in the air. The gray parts in the diagrams represent the stance phases, while the white parts represent the swing phases.

TABLE III
DISTANCE COMPARISON UNDER DIFFERENT GAITS

Morphology \ Gait	PSO	Ours	Walk	Trot	Pace
$[\xi_1^*, \xi_2^*, \xi_3^*, \xi_4^*, \xi_5^*, \xi_6^*]$	411.27	425.52	351.43	379.52	63.85

more effective and reliable interactive tuples stored in the replay buffer. The introduction of the CPG modules simplifies the optimization problem and improves training speed.

2) Compared With Gaits Optimized by PSO: In this section, we select the random seed that yields the best performance out of the five seeds and utilize the robot's morphology parameters obtained from the final round of morphology optimization (represented by $[\xi_1^*, \xi_2^*, \xi_3^*, \xi_4^*, \xi_5^*,$ and $\xi_6^*]$). We employ the PSO method and the proposed gait optimization method to optimize the robot's gait parameters. The optimized gait parameters are then used to measure the robot's movement distance under predefined steps, and the results are illustrated in Table III. We employ 30 particles and 100 iterations to strike a balance between effectiveness and efficiency.

Table III shows that the results obtained by the PSO optimization method are comparable to those acquired by the proposed method, indicating that both methods can yield optimal gait parameters. However, during the upper-level optimization, the PSO cannot directly provide an optimal strategy for calculating fitness for each morphology candidate. Instead, the PSO needs to be reoptimized for each new morphology parameter in order to obtain a feasible strategy and utilize it to calculate fitness. This undoubtedly decreases algorithm efficiency.

3) Compared With Classical Gaits: Quadruped gait refers to the walking mode with a fixed phase relationship between legs. For quadrupeds, there are mainly the following kinds of gaits. 1) The first is the walk gait, also known as wave gait, each foot rises and falls in turn, and the phase difference is a quarter cycle. 2) The second is the trot gait, also known as the diagonal gait, in which the diagonal legs rise and fall in pairs with a phase difference of half-cycle between the pairs. 3) The next is the pace gait, in which the legs on the same side rise and fall in pairs with a phase difference of half-cycle between the pair [31]. Generally, for the velocity level of quadrupeds, the walk gait is the lowest, the trot gait is higher than the former, and the pace gait is seen as similar to the trot ones.

To compare the trained gaits with classical gaits, we assign the robot morphology parameters as used in Section V-C2 and compare the distance traveled by different gaits under the predefined steps. As demonstrated in Table III, among the three classical gaits, the trot gait produces the fastest movement, followed by the walk gait and then the pace gait. The trained gait travels the farthest distance, demonstrating the effectiveness of combining DRL with the CPG to learn gaits for limited DoAs legged robots. To investigate the relationship between the trained gait and the three classical gaits, we plot the corresponding phase diagrams in Fig. 8. It is evident that the trained gait exhibits a small phase difference between the FL leg and HR leg, similar to the trot gait. The other two legs maintain a quarter-phase difference from the former two, much like the walk gait. The leg's workspace of a four DoAs legged robot is much smaller than that of a typical twelve DoAs legged robot, which has three joints per leg. As a result, there are significant differences in their locomotion capabilities. For example, although the pace gait is expected to have a velocity similar to the trot gait in nature, our experiments demonstrate that the pace gait has the slowest velocity for the four DoAs legged robots. In the constrained workspace, previous knowledge of gait patterns for twelve DoAs legged robots may not be applicable. In the proposed method, the four DoAs legged robot interacts with the environment through trial and error to maximize rewards. This approach takes into account the constraints imposed by the limited DoAs, resulting in improved outcomes. It is worth noting that the learning-based method is influenced by the random seed, resulting in slight variations between the optimized morphology parameters and their corresponding gait parameters. These values are not unique, and in our experiments, we present the results for the random seeds that yielded the best outcomes.

VI. PHYSICAL EXPERIMENTS RESULTS

In the previous section, we analyze the experimental results and discover that, for the majority of gaits, four-legged robots

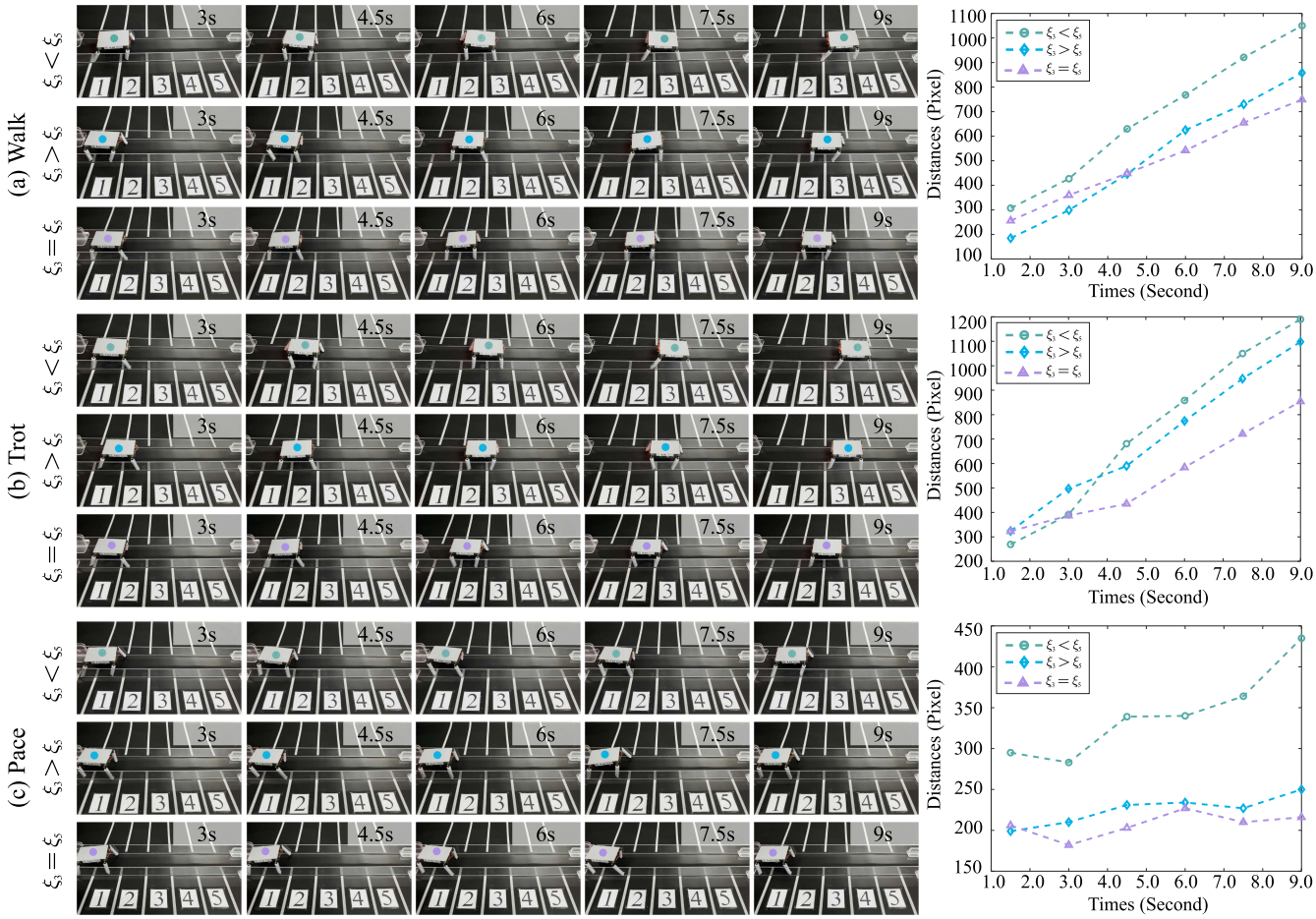

Fig. 9. Comparison of asynchronous states at the same moment of physical robots. (a) Represents walk gaits. (b) Represents trot gaits. (c) Represents pace gaits. Each group is further divided into three rows: the first row depicts a front leg shorter than the hind leg ($\xi_3 < \xi_5$), the second row depicts a front leg longer than the hind leg ($\xi_3 > \xi_5$), and the third row depicts equal-length front and hind legs ($\xi_3 = \xi_5$). To facilitate clear visualization of the distance traveled by the robot, we have marked the numbers below each robot from left to right as 1, 2, 3, 4, and 5, where the distance between each marker is 10 cm. On the right-hand side of each row, line graphs illustrate the position of the geometric center of the robot, representing the trend of the robot's movement at each moment.

with limited DoAs run faster when the front legs are shorter than the hind legs. Furthermore, we compare our gait optimization method with classical gaits and illustrate that the trained gait can improve the robot's running speed. In this section, we further validate these observations using physical robots.

A. Morphology Optimization

In Section V-B, it is observed that longer hind legs enable four legged robots with limited DoAs to achieve higher speed in most gaits. To validate this conclusion through physical experiments, we design three sets of morphology parameters for three classical gaits. For the first set ($\xi_3 < \xi_5$), we use the optimized morphology parameters previously used in Section V-C2. For the second set ($\xi_3 > \xi_5$), we exchange the position of ξ_3^* and ξ_5^*, while keeping the remaining parameters fixed. For the third set ($\xi_3 \leq \xi_5$), we replace ξ_5^* with ξ_3^* in the morphology parameter.

To visually present the positions attained by each robot at identical time intervals, we photograph and depict them in Fig. 9. The corresponding times taken for each gait are provided in Table IV. It is evident from Fig. 9 that the robot moves fastest when the front legs are shorter than the hind legs, followed by front legs longer than the hind legs, and finally equal-length front and hind legs. These results align with the findings of the simulation experiments.

TABLE IV
TIME REQUIRED TO REACH THE END USING DIFFERENT GAITS UNDER THREE RATIOS OF LEG LENGTHS

Morphology	Walk	Trot	Pace
$\xi_3 < \xi 5$	13 s	11 s	80 s
$\xi_3 > \xi 5$	18 s	15 s	150 s
$\xi_3 = \xi 5$	24 s	20 s	200 s

B. Gait Optimization

We apply the optimized results from gait optimization to real physical robots, where the main difference is that the real robots lack sensors. In the simulation environment, sensor data is used

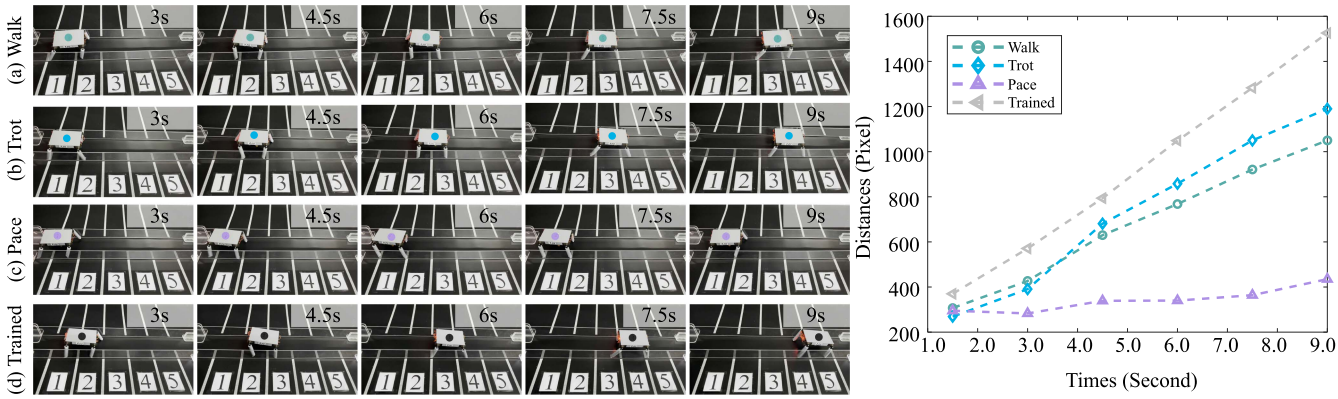

Fig. 10. Comparison of asynchronous states at the same moment of physical robots. These are the results obtained by different gaits in the optimal morphology parameters. The first row pertains to the walk gait, the second row represents the trot gait, the third row is the pace gait, and the final row is the trained gait.

to train the policy by providing state information as input to the algorithm. Once the policy converges, we observe that the robot's optimal gait approaches a stable value on horizontal ground. To directly transfer this optimal gait to the real robots, we record the corresponding optimal gait of the simulated four DoAs legged robots and program the position control data onto the real robots. The real robots are controlled using pulse width modulation (PWM). We conduct experiments on the real robots using the recorded optimal gait, as well as three classical gaits (walk, trot, and pace), under the optimized morphological parameters. Fig. 10 shows the results of our experiments, with each line representing a different gait. It is apparent that the trained gait is the fastest, followed by the trot, walk, and pace gaits, respectively. Both the trot and walk gaits reached the position marked between 3 and 4 within the recording timeframe, while the trained gait reached position 5, and the pace gait only reached position 1. These findings align with the results obtained in the simulation.

VII. Conclusion

In this article, we propose a co-optimization method for small-scale legged robots that have limited DoAs to enhance their mobile performance while reducing energy consumption. We model co-optimization as a bi-level optimization problem, with the lower level optimizing gait and the upper level optimizing morphology parameters. In the simulation, we compare the results of upper-level optimization and lower-level optimization with comparison methods and analyze the outcomes. The upper-level optimization reveals that four DoAs legged robots run faster when the front legs are shorter than the hind legs in most gaits. Furthermore, through lower-level optimization, we discover that trained gaits run faster than classical gaits, and when the DRL algorithm converges, the optimal CPG parameters approach a constant value, allowing the optimized gait to be directly transplanted into the physical environment. In deploying physical experiments, we confirm the findings of the simulation experiments hold true in real-life scenarios. Our future work will concentrate on further optimizing physical robots and installing sensors to establish a closed-loop control system, allowing them to adapt to changing environments such as locomotion in uneven terrain, obstacles, variations in friction, etc.. Furthermore, we plan to extend the current approach to identify optimized parameters for more complex robot configurations.

References

[1] C. Huang, Z. Lai, X. Wu, and T. Xu, "Multimodal locomotion and cargo transportation of magnetically actuated quadruped soft microrobots," *Cyborg Bionic Syst.*, vol. 2022, 2022, Art. no. 0004.

[2] C. T. Nguyen et al., "Printable monolithic hexapod robot driven by soft actuator," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2015, pp. 4484–4489.

[3] M. Rubenstein, C. Ahler, and R. Nagpal, "Kilobot: A low cost scalable robot system for collective behaviors," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2012, pp. 3293–3298.

[4] A. M. Hoover, E. Steltz, and R. S. Fearing, "Roach: An autonomous 2.4 G crawling hexapod robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2008, pp. 26–33.

[5] A. G. Dharmawan, H. H. Hariri, S. Foong, G. S. Soh, and K. L. Wood, "Steerable miniature legged robot driven by a single piezoelectric bending unimorph actuator," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2017, pp. 6008–6013.

[6] H. Xie, Z. Gao, G. Jia, S. Shimoda, and Q. Shi, "Learning rat-like behavioral interaction using a small-scale robotic rat," *Cyborg Bionic Syst.*, vol. 4, 2023, Art. no. 0032.

[7] L. Wang et al., "Design and dynamic locomotion control of quadruped robot with perception-less terrain adaptation," *Cyborg Bionic Syst.*, vol. 2022, 2022, Art. no. 9816495.

[8] S. Ha, S. Coros, A. Alspach, J. Kim, and K. Yamane, "Joint optimization of robot design and motion parameters using the implicit function theorem," in *Robotics: Science and Systems*, Cambridge, MA, USA: MIT Press, 2017.

[9] M. Geilinger, R. Poranne, R. Desai, B. Thomaszewski, and S. Coros, "SkaterBots: Optimization-based design and motion synthesis for robotic creatures with legs and wheels," *ACM Trans. Graph.*, vol. 37, no. 4, pp. 1–12, 2018.

[10] M. Geilinger, S. Winberg, and S. Coros, "A computational framework for designing skilled legged-wheeled robots," *IEEE Robot. Automat. Lett.*, vol. 5, no. 2, pp. 3674–3681, Apr. 2020.

[11] G. Fadini, T. Flayols, A. Del Prete, N. Mansard, and P. Souéres, "Computational design of energy-efficient legged robots: Optimizing for size and actuators," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 9898–9904.

[12] D. J. Hejna III, P. Abbeel, and L. Pinto, "Task-agnostic morphology evolution," in *Proc. Int. Conf. Learn. Representations*, 2020.

[13] T. Wang, Y. Zhou, S. Fidler, and J. Ba, "Neural graph evolution: Towards efficient automatic robot design," in *Proc. Int. Conf. Learn. Representations*, 2018.

[14] A. Gupta, S. Savarese, S. Ganguli, and L. Fei-Fei, "Embodied intelligence via learning and evolution," *Nature Commun.*, vol. 12, no. 1, p. 5721, 2021.
[15] D. Ha, "Reinforcement learning for improving agent design," *Artif. Life*, vol. 25, no. 4, pp. 352–365, 2019.
[16] S. Hu, Z. Yang, and G. Mori, "Neural fidelity warping for efficient robot morphology design," in *Proc. IEEE Int. Conf. Robot. Automat.*, 2021, pp. 7079–7086.
[17] K. S. Luck, H. B. Amor, and R. Calandra, "Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning," in *Proc. Conf. Robot Learn.*, 2020, pp. 854–869.
[18] C. Schaff, D. Yunis, A. Chakrabarti, and M. R. Walter, "Jointly learning to construct and control agents using deep reinforcement learning," in *Proc. Int. Conf. Robot. Automat.*, 2019, pp. 9798–9805.
[19] G. Brockman et al., "Openai gym," 2016, *arXiv:1606.01540*.
[20] C. Wang, G. Xie, L. Wang, and M. Cao, "CPG-based locomotion control of a robotic fish: Using linear oscillators and reducing control parameters via PSO," *Int. J. Innov. Comput. Inf. Control*, vol. 7, no. 7B, pp. 4237–4249, 2011.
[21] L. Li, C. Wang, and G. Xie, "A general CPG network and its implementation on the microcontroller," *Neurocomputing*, vol. 167, pp. 299–305, 2015.
[22] S. Yan, Z. Wu, J. Wang, M. Tan, and J. Yu, "Efficient cooperative structured control for a multi-joint biomimetic robotic fish," *IEEE/ASME Trans. Mechatron.*, vol. 26, no. 5, pp. 2506–2516, Oct. 2021.
[23] X. Liu, R. Gasoto, Z. Jiang, C. Onal, and J. Fu, "Learning to locomote with artificial neural-network and CPG-based control in a soft snake robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst.*, 2020, pp. 7758–7765.
[24] A. Crespi, D. Lachat, A. Pasquier, and A. J. Ijspeert, "Controlling swimming and crawling in a fish robot using a central pattern generator," *Auton. Robots*, vol. 25, no. 1, pp. 3–13, 2008.
[25] T. Haarnoja et al., "Soft actor-critic algorithms and applications," 2018, *arXiv:1812.05905*.
[26] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," in *Proc. Int. Conf. Mach. Learn.*, 2018, pp. 1861–1870.
[27] F. Nogueira, "Bayesian optimization: Open source constrained global optimization tool for Python," 2014. [Online]. Available: https://github.com/fmfn/BayesianOptimization
[28] A. Fukuhara, M. Gunji, Y. Masuda, K. Tadakuma, and A. Ishiguro, "Simulation study on galloping quadruped robot with flexible shoulder Hammock structure," in *Proc. Int. Symp. Adaptive Motion Animals Machines*, 2021, pp. 32–33.
[29] H. Zou and J. P. Schmiedeler, "The effect of asymmetrical body-mass distribution on the stability and dynamics of quadruped bounding," *IEEE Trans. Robot.*, vol. 22, no. 4, pp. 711–723, Aug. 2006.
[30] Y. Liu et al., "Complex three-dimensional terrains traversal of insect-scale soft robot," *Soft Robot.*, vol. 10, no. 3, pp. 612–623, 2023.
[31] D. F. Hoyt and C. R. Taylor, "Gait and the energetics of locomotion in horses," *Nature*, vol. 292, no. 5820, pp. 239–240, 1981.

Ci Chen received the B.E. degree in agricultural mechanization and automation from Northeast Agricultural University, Harbin, China, in 2018. She is currently working toward the Ph.D. degree in control science and engineering with the Institute of Cyber-Systems and Control, Department of Control Science and Engineering, Zhejiang University, Hangzhou, China.

Her research interests include deep reinforcement learning and legged robots.

Pingyu Xiang received the B.E. degree in mechatronic engineering from the School of Mechanical Engineering, Zhejiang University, Hangzhou, China, where he is currently working toward the master's degree in control science and engineering with the College of Control Science and Engineering, under the supervision of Prof. Lu.

His research interests continuum robotics, medical robots, and machine learning.

Jingyu Zhang received the B.E. degree in mechatronic engineering from the Nanjing University of Science and Technology, Nanjing, Jiangsu, China. He is currently working toward the Ph.D. degree in robotics with the Soft Medical Robotics Lab, Zhejiang University, Hangzhou, China.

He has previously worked on medical ultrasound scanning systems and is currently developing novel different scale of soft continuum to augment the operation ability of surgeons.

Rong Xiong (Senior Member, IEEE) received the Ph.D. degree in control science and engineering from the Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2009.

She is currently a Professor with the Department of Control Science and Engineering, Zhejiang University. Her current research interests include motion planning and SLAM.

Yue Wang (Member, IEEE) received the Ph.D. degree in control science and engineering from the Department of Control Science and Engineering, Zhejiang University, Hangzhou, China, in 2016.

He is currently an Associate Professor with the Department of Control Science and Engineering, Zhejiang University. His current research interests include mobile robotics and robot perception.

Haojian Lu (Member, IEEE) received the B.E. degree in mechatronic engineering from the Beijing Institute of Technology, Beijing, China, in 2015, and the Ph.D. degree in robotics from the City University of Hong Kong, Kowloon, Hong Kong, in 2019.

He is currently a Professor with the State Key Laboratory of Industrial Control and Technology, and the Institute of Cyber-Systems and Control, Zhejiang University, Hangzhou, China. His research interests include micro/nanorobotics, bioinspired robotics, medical robotics, micro aerial vehicles, and soft robotics.