IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Date of publication xxxx 00, 0000, date of current version xxxx 00, 0000.

Digital Object Identifier xx.xxxx/ACCESS.xxxx.DOI

Reinforcement Learning based Cascade Motion Policy Design for Robust 3D Bipedal Locomotion

GUILLERMO A. CASTILLO[1], (Student Member, IEEE), BOWEN WENG[1], (Student Member, IEEE), WEI ZHANG[2], (Senior Member, IEEE), and AYONGA HEREID[3], (Member, IEEE)

[1]Electrical and Computer Engineering Department, The Ohio State University, Columbus, OH 43210 USA (e-mail: castillomartinez.2@osu.edu, weng.172@osu.edu)
[2]SUSTech Institute of Robotics, Southern University of Science and Technology (SUSTech), China (e-mail: zhangw3@sustech.edu.cn)
[3]Mechanical and Aerospace Engineering Department, The Ohio State University, Columbus, OH 43210 US (e-mail: hereid.1@osu.edu)

Corresponding author: Guillermo A. Castillo (e-mail: castillomartinez.2@osu.edu).

This work was supported in part by the OSU M&MS Discovery Theme Initiative, and the National Natural Science Foundation of China under Grant No. 62073159.

ABSTRACT This paper presents a novel reinforcement learning (RL) framework to design cascade feedback control policies for 3D bipedal locomotion. Existing RL algorithms are often trained in an end-to-end manner or rely on prior knowledge of some reference joint or task space trajectories. Unlike these studies, we propose a policy structure that decouples the bipedal locomotion problem into two modules that incorporate the physical insights from the nature of the walking dynamics and the well-established Hybrid Zero Dynamics approach for 3D bipedal walking. As a result, the overall RL framework has several key advantages, including lightweight network structure, sample efficiency, and less dependence on prior knowledge. The proposed solution learns stable and robust walking gaits from scratch and allows the controller to realize omnidirectional walking with accurate tracking of the desired velocity and heading angle. The learned policies also perform robustly against various adversarial forces applied to the torso and walking blindly on a series of challenging and unstructured terrains. These results demonstrate that the proposed cascade feedback control policy is suitable for navigation of 3D bipedal robots in indoor and outdoor environments.

INDEX TERMS Motion control, Legged locomotion, Machine learning

I. INTRODUCTION

WHILE human and biological bipeds can naturally learn complex motion planning, it is still a challenging task for bipedal robots due to the highly unstable nature of bipedal robots. Properties like underactuation, unilateral ground contacts and impacts, nonlinear dynamics, and high degrees of freedom significantly increase the complexity of synthesizing feasible robot motions. Various learning-based solutions, especially with the recent progress on deep learning, have shown remarkable performance in solving challenging control problems in bipedal locomotion. In general, these learning-based approaches can be further classified into end-to-end methods, and reference trajectory learning approaches.

Bipedal locomotion's most common learning objective is a feedback control policy that directly maps the state inputs to the torque control output or the joint angles. Typically, this policy is constructed in an end-to-end manner, and the learned policy serves the general purpose of stability maintenance (i.e., walking without falling). Various learning methods have shown effectiveness in learning an end-to-end control policy. Policy gradient based approaches such as DDPG and PPO have demonstrated competitive performance for general robotic locomotion tasks in simulations with end-to-end learning using torque output policies [1], [2] and real-world experiments (typically combined with dynamics randomization) using torque output-based end-to-end learning [3], [4] and joint angle-based end-to-end learning [5], [6].

Some more advanced methods also seek to achieve velocity tracking [7], push-recovery [8], and walking in various terrain conditions [9] through more structured frameworks. The velocity tracking policy from [7] relies on prior knowl-

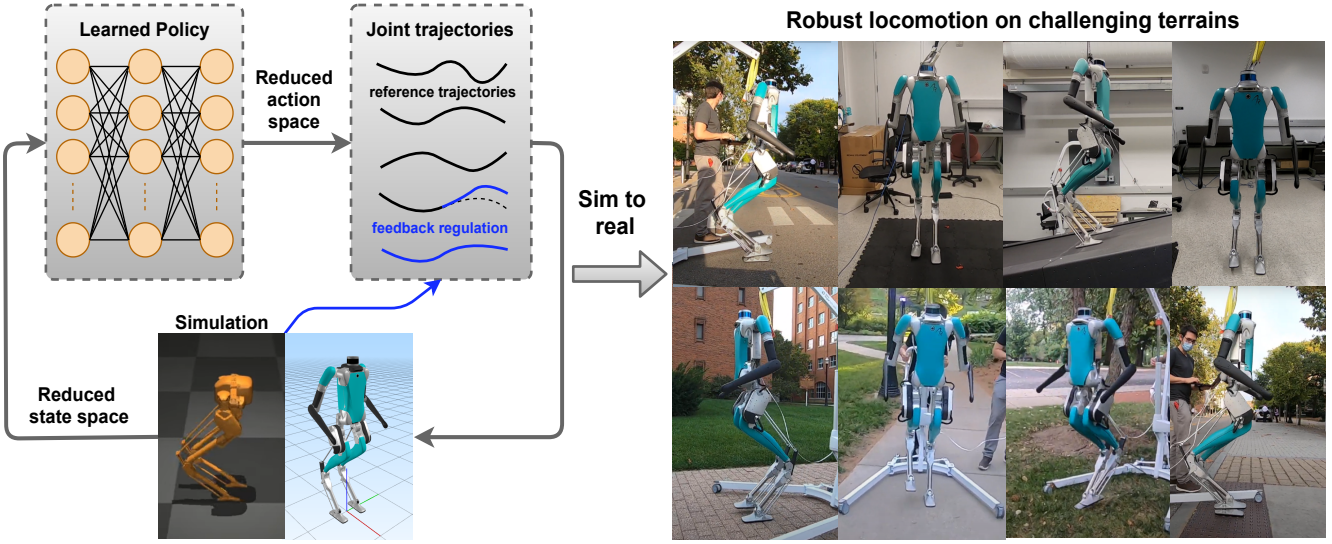

FIGURE 1: Overview of the proposed learning framework. A cascade controller combines the RL motion planning module with the feedback regulator module to realize stable and robust locomotion. The learned policy is successfully transferred to hardware, allowing the Digit robot to walk on different challenging terrains using the same policy. An overview video of all the experimental results can be found at https://sites.google.com/view/rl-cmpd.

edge of a good joint reference trajectory and only learns small compensations added to the known reference trajectory. Siekmann et al. proposed to combine PPO with recurrent neural network (RNN) for learning the direct control policy for Cassie [10]. Some also extend the deep reinforcement learning approach with provided guidance for motion mimicking [11], [12].

Another learning objective is to acquire a reference tracking trajectory of a selected anchor point (e.g., the center of gravity point of the upper torso). A lower-level controller then seeks to track such a learned reference through basic model information such as kinematics. Morimoto et.al. [13], [14] learned the Poincare map of the periodic walking pattern and applied the method to two 2D bipedal robots. Some recent work has proposed to learn the joint-level trajectory as the reference motion through supervised learning [15] and reinforcement learning [16]–[18]. The authors in [19] learn linear policies that map the reduced robot's state to parameterized elliptical trajectories for the robot's feet. These approaches often simplify the design of the lower-level tracking, which can be as simple as a PD controller.

Despite the empirical success, most of the aforementioned learning-based approaches are sampling inefficient (millions of data samples) and are usually over-parameterized (thousands of tunable parameters). It is also worth emphasizing that the reference-trajectory-learning approach makes it easier to induce gait symmetry and smooth control signals within the bounded admissible space. On the other hand, the end-to-end approach is difficult to handle symmetry and torque constraints, hence may lead to unnatural walking gaits and sparky control signals [20].

In this work, we propose a trajectory-based RL framework to address some of the challenges found in the learning of bipedal locomotion. By decoupling the problem of bipedal locomotion as a two-phase process: trajectory planning and feedback regulation, we propose a modular solution that incorporates the physical insights of dynamic locomotion and its hybrid nature into the learning process of the policy. In particular, we leverage the exploration potential of RL algorithms to find reference trajectories for dynamic locomotion using a reduced state of the robot. Then, we improve these reference trajectories using feedback regulation to obtain stable and robust walking gaits. This decoupled structure significantly simplifies the neural network's complexity, enhancing sampling efficiency and robustness of the learned policy.

A method similar to ours is presented in [21], where the authors propose a decoupled structure that uses DRL to learn a Finite State Machine (FSM) based policy that outputs reference trajectories for particular joints of the robot. A simple linear balance feedback controller is then used on top of the reference trajectories to produce robust locomotion. In our proposed work, we compute continuous joint-space trajectories by means of 5th-order Bézier Polynomials. In addition, we use different high-level commands, e.g., desired velocity tracking, as part of the reduced-order state of our learning framework, whereas [21] uses the full-order state of the robot in addition to desired gait parameters: step length, step duration, and maximum swing foot height during a step.

Our proposed method is evaluated with different robot models, including simulation of the bipedal robots Rabbit, Cassie, and Digit. In addition, we show that the proposed controller structure can be used to transfer the learned policy successfully to hardware with minimal tuning. The resulting

controller is extensively tested in hardware with the Digit robot, showing effective velocity tracking performance, and robustness to different disturbances such as external adversarial forces and uneven terrains.

Preliminary results of this work were presented in conference papers [18], [22]. In this paper, we extend the preliminary results to further increase the efficiency of the learning method, consider an additional degree of freedom to include constrained arm's motion into the walking gait, include additional regulations to improve the performance of the controller, and perform an extensive series of indoors and outdoors experiments to demonstrate the good performance of the learned policy on hardware. Our contribution can be summarized as follows:

- We propose a complete RL framework to learn robust and stable walking gaits from scratch for 3D bipedal robots. The method takes insights from the hybrid and symmetric nature of dynamic walking to significantly reduce the state and action spaces of the policy, enhancing the sample efficiency of the learning process and robustness of the walking gait.
- We design a regulator policy that uses simple but effective feedback regulators to improve the stability and robustness of the learned walking gait. Different from the earlier conference version, we also develop an estimator of the terrain slope to improve the swing foot orientation regulator, which is the key to successful outdoor experiments. Moreover, we add a stance foot regulation that facilitates velocity tracking on hardware.
- We demonstrate that the proposed framework can be easily extended to robots with different DoF and morphology. We use the proposed learning framework to control the bipedal robot Cassie (no arm joints) and the humanoid robot Digit (with arm joints). The results show the same RL framework learns stable walking gaits for both robots. The results have also been validated extensively in both simulation and hardware.
- We conduct extensive experiments to test the performance of the policy on real hardware, demonstrating the learned policy has a good tracking performance on the desired waking velocity and the desired torso orientation. These results enable the application of the proposed RL framework with confidence for terrain navigation in indoor and outdoor environments. Most of the learning frameworks for bipedal locomotion proposed in the literature do not provide details about the performance of the learned policy for tracking high-level commands like the torso's desired velocity and orientation.

The remainder of the paper is organized as follows. Section II introduces the problem of bipedal locomotion and its formulation as a cascade motion control framework. In Section III we present the motion policy design as a RL problem with a reduced state and action spaces. Section IV introduces the design of the feedback regulator policy used to convert the joint action commands into admissible torques applied to the joints. In Section V, we show the details of the application of the proposed framework to two different bipedal robots, Cassie and Digit, and Section VI presents the simulation and hardware results. Finally, Section VII provides concluding remarks about this work.

II. PRELIMINARIES AND PROBLEM FORMULATION

A. BIPEDAL ROBOT MODEL

Bipedal locomotion consists of a collection of phases of continuous dynamics with discrete events that trigger the transitions between these continuous dynamics phases; formally, modeling both continuous and discrete dynamics together results in a hybrid system model. The configuration space $\mathcal{Q}$ of a robot can typically be represented by a floating-base generalized coordinate system, defined as

$$q = [p_b, \phi_b, q_r] \in \mathcal{Q}, \tag{1}$$

where $p_b = (q_x, q_y, q_z) \in \mathbb{R}^3$ denotes the relative position of the robot's base, $\phi_b \in SO(3)$ denotes the orientation of the robot's base frame, and $q_r \in \mathbb{R}^m$ denotes the relative angles of articulated joints. Throughout this paper, we use $\dot{p}_b = (v_x, v_y, v_z)$ to represent the velocity of the robot's frame, $\phi_b = (q_\psi, q_\theta, q_\phi)$ as the Euler's angle representation (roll, pitch, yaw) of the robot's base orientation, and $\dot{\phi}_b = (\dot{q}_\psi, \dot{q}_\theta, \dot{q}_\phi)$ represents the angular velocity of the robot's base.

Letting $x = (q, \dot{q}) \in \mathcal{X}$ denote the robot states, $u \in \mathcal{U} \subseteq \mathbb{R}^m$ a vector of actuator inputs, and $\omega \in \Omega \subseteq \mathbb{R}^w$ a vector of disturbances and uncertainties, the hybrid system model for bipedal locomotion can be defined as

$$\Sigma : \begin{cases} \dot{x} &= f(x, u; \omega) & x \notin \mathcal{D} \\ x^+ &= \Delta(x^-) & x^- \in \mathcal{D}, \end{cases} \tag{2}$$

where, f represents the continuous dynamics. The switching surface $\mathcal{D}$ is typically the (hyper-) surface of points corresponding to the height of the swing leg above the ground being zero, and $\Delta : \mathcal{D} \to \mathcal{X}$, the reset map or impact map [23], determines the post-impact state values x^+ just after switching as a function of the pre-impact state values x^- just before switching.

B. BIPEDAL LOCOMOTION PROBLEM

In general, the bipedal locomotion problem seeks to establish a motion control policy $\pi : \mathcal{X} \times \mathcal{C} \to \mathcal{U}$ with $\mathcal{C}$ being a set of high-level locomotion commands, such that some properties are achieved. For example, the desired properties may include (i) following commands, (ii) maintaining feasibility condition, (iii) satisfying admissibility condition, (iv) exhibiting naturalistic locomotion, and (v) robustness against uncertainties and disturbances. Here, we mathematically define the aforementioned properties as follows to define the bipedal locomotion problem formally.

Following Command. We would like the robot to follow specific high-level commands, such as desired velocities or

target locations. In this paper, we are particularly interested in velocity tracking, which can be defined as the asymptotic convergence to the desired velocity profile $v^d(t)$, given as,

$$\lim_{t\to\infty} \left\| \bar{v}(x) - v^d(t) \right\| = 0, \tag{3}$$

where $\bar{v}(x)$ denotes the average velocity over a walking step.

State Feasibility Condition. Let $\mathcal{Z} \subseteq \mathcal{X}$ be a set of forbidden states that are prohibitive for the robot. Hence the feasibility criterion is equivalent to ensuring the set $\mathcal{X}^* = \mathcal{X} \setminus \mathcal{Z}$ forward invariant given the dynamics model (2), i.e.,

$$\forall x(0) \in \mathcal{X}^* \rightarrow x(t) \in \mathcal{X}^*, \quad \forall t \geq 0. \tag{4}$$

Input Admissibility Condition. Let $\mathcal{U}^*$ be the nominal admissible actuator input set of the robot determined by the actuators' physical capability. The admissibility criterion requires the actuator inputs are persistently feasible, i.e.,

$$\pi(x, c) \in \mathcal{U}^* \quad \forall x \in \mathcal{X}^*, c \in \mathcal{C}. \tag{5}$$

Naturalistic Locomotion. Moreover, the bipedal applications also expect naturalistic motion for various causes (e.g., environment adaptation and energy efficiency). Examples of naturalistic motion include maintaining the upper-body straight, the Center-of-Mass (CoM) within the support polygon described by the robot feet, avoiding the collision of the robot's feet with each other, etc. In this paper, we consider the torso angle limits and the constrained Center-of-Mass (CoM) position to characterize the naturalistic behavior. In particular, let $\theta_{\text{tor}}(x) : \mathcal{X} \rightarrow SO(3)$ represents the orientation of the robot's torso, the following constraint is expected to be satisfied:

$$\theta_{\text{tor}}(x) \in \Theta, \quad \forall x \in \mathcal{X}, \tag{6}$$

where $\Theta \in SO(3)$ represents the admissible range for the roll, yaw and pitch angles of the robot's torso. In addition, let $p_{\text{com}} : \mathcal{X} \rightarrow \mathbb{R}^3$ be the CoM position with respect to the stance foot in the Cartesian coordinate. The following condition confines the projection of CoM within a enclosed region determined by both feet and the height of CoM within a certain threshold:

$$p_{\text{com}}(x) \in P \quad \forall x \in \mathcal{X}, \tag{7}$$

where $P \subset \mathbb{R}^3$ represents the admissible CoM range.

Problem 1: [**The Bipedal Locomotion Problem**] Consider the robot model in (2), the Bipedal Locomotion Problem seeks to establish a motion control policy $\pi : \mathcal{X} \times \mathcal{C} \rightarrow \mathcal{U}$, such that the criterion defined in (3) - (7) are satisfied with the presence of model uncertainty and external disturbance.

In practice, solving the above problem is challenging as the hybrid dynamical system in (2) is too complex to have a model-based solution that guarantees the satisfaction of all desired properties. Moreover, the various properties specified cannot be satisfied simultaneously in principle (e.g., the velocity tracking requirement may be relaxed in exchange for the safety assurance). In this paper, we propose to solve the problem using a cascaded structure that combines the reinforcement learning (RL) based motion planning and model-based feedback control design.

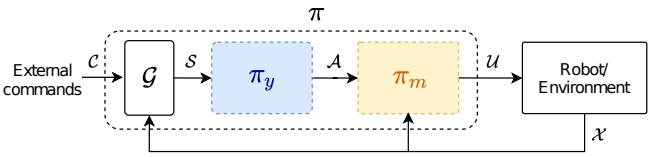

FIGURE 2: The cascaded structure of the proposed motion control policy framework for bipedal locomotion.

C. CASCADED MOTION CONTROL FRAMEWORK

Our proposed approach takes inspiration from the generalized Hybrid Zero Dynamics (G-HZD) framework presented in [24], [25]. As shown in Figure 2, the motion control policy π in Problem 1 consists of a feature selection module, $\mathcal{G}$, and two cascaded policies: a motion policy π_y and a feedback control policy π_m. To clearly identify the objectives of this paper, we formally define the proposed motion control framework as follows, where the design of each component will be presented in detail in the following sections.

Problem 2: [**Cascade Motion Control Policy Design**] The motion control policy π in Problem 1 can be designed as

$$\pi = \pi_m(\cdot) \circ \pi_y(\cdot) \circ \mathcal{G}(\cdot). \tag{8}$$

The feature selection module $\mathcal{G} : \mathcal{X} \times \mathcal{C} \rightarrow \mathcal{S}$ maps the full-order states and external commands to a reduced-dimensional feature states $s \in \mathcal{S}$. The motion policy $\pi_y(\cdot) : \mathcal{S} \rightarrow \mathcal{A}$ will be designed to generate feasible joint actions $\alpha \in \mathcal{A}$, with $\mathcal{A}$ being the action space, that satisfy the conditions defined in (3) - (7). Finally, the feedback control policy $\pi_m(\cdot) : \mathcal{X} \times \mathcal{A} \rightarrow \mathcal{U}$ converts the joint action commands to admissible actuator inputs with the objective of keeping the robot from falling and simultaneously satisfying (3) - (7).

While there are various ways to design the motion policy for bipedal locomotion in literature, our work particularly focuses on reinforcement learning (RL) design approaches [6], [10], [11]. Despite recent success of RL-based approaches in robust sim-to-real transfer of the policy on robot hardware, existing approaches still suffer from sampling inefficiency and often requires prior knowledge of good reference trajectories in training [10], [11]. The proposed trajectory-based RL motion policy design (see Section III) aims to tackle existing limitations of RL-based approaches in bipedal locomotion by incorporating insights from model-based control methods with data-driven reinforcement learning to realize robust bipedal locomotion policies. In addition, an intuitive feedback regulation controller policy (see Section IV) is designed to improve the overall robustness of the motion policy.

Remark 1: A classic end-to-end RL solution to the bipedal locomotion problem can be considered as a special case of Problem 2. Instead of using the decoupled structure, the end-to-end approaches train a single neural network (NN) policy

$\pi(\cdot) : \mathcal{X} \to \mathcal{U}$ that maps the full order states directly to the actuator inputs. However, this approach blindly uses all the data available without insights about the nature or structure of the bipedal locomotion problem, resulting in largely inefficient training with learned policies that are not feasible to be implemented safely in hardware [1].

III. MOTION POLICY DESIGN

In this section, we present a sample efficient RL framework for the motion policy design problem described in Section II. The overall structure of the proposed RL-based cascade motion policy is presented in Figure 3. We will start with the formal definition of the RL framework for our later discussion. Then we will comprehensively discuss the design of reduced state and action spaces and the specific learning procedure for bipedal locomotion.

A. REINFORCEMENT LEARNING FRAMEWORK

A typical reinforcement learning approach considers a Markov Decision Process (MDP) as a tuple of components, defined as

$$\mathcal{M} := (\mathcal{S}, \mathcal{A}, \mathsf{P}, r, \xi, \gamma). \tag{9}$$

Here $\mathcal{S}$ is the feature state space, and $\mathcal{A}$ is the feasible action space. Specifically, given $s_t \in \mathcal{S}$ at time t, an agent (i.e., the motion planner) takes an action $\alpha_t \in \mathcal{A}$, transits into the next feature state $s_{t+1} \in \mathcal{S}$ according to the transition probability $\mathsf{P}(s_{t+1}|s_t, \alpha_t)$ and receives a reward $r(s_t, \alpha_t, s_{t+1})$. Moreover, ξ denotes the distribution of the initial state $s_0 \in \mathcal{S}$ and $\gamma \in (0, 1)$ denotes the discount factor. The goal of the reinforcement learning (RL) framework is to find an optimal motion policy $\pi^* : \mathcal{S} \to \mathcal{A}$ that maximizes a long-term accumulated reward, defined as

$$J(\pi) = (1-\gamma)\mathbb{E}[\sum_{t=0}^{\infty} \gamma^t r(s_t, \alpha_t, s_{t+1})]. \tag{10}$$

To cast bipedal motion policy as a RL problem, one requires (i) adapting the model (2) to the MDP form of (9), and (ii) configuring the criterion in Problem 1 to align with the RL settings. It is immediate that the probabilistic transition part in (9) is equivalent to the described bipedal robot model (2). The stochastic transition of the MDP process captures the disturbances and uncertainties ω such as the random sampling of initial states in the policy training and dynamics uncertainty due to the random interactions with the environment (e.g., early or late ground impacts). Moreover, the desired properties in Problem 1 can be either characterized as rewards or hard constraints in RL. In this paper, we formulate criterion (3), (5), (6), (7) as rewards, and (4) as hard constraints.

B. STATE SPACE

In our proposed framework, a neural-network motion policy π_y maps a feature $s \in \mathcal{S}$ to an action $\alpha \in \mathcal{A}$ via a probability distribution $\pi_y(\cdot|s)$. In particular, the feature can be decomposed into an *endogenous* component, ζ, and an *exogenous* component, η. The endogenous component ζ is a reduced dimensional representation of the robot states. The exogenous component η corresponds to the external commands, such as desired walking speeds or turning directions, terrain slope, whose transitions will not be affected by the agent through actions [26]. The inclusion of exogenous components enables a single motion policy to capture various locomotion tasks and smooth transitions among these tasks.

Reduced Dimensional Feature Representation. Many existing learning-based approaches for bipedal locomotion use the full-order state as the input of the neural network policy, which significantly reduces the sampling efficiency of the training process, resulting in unnecessarily large neural networks and prolonged training time. In this paper, we take inspiration from classic model-based approaches in bipedal locomotion to design a lightweight neural network policy structure to improve sampling efficiency and reduce the training time. In particular, we choose as a reduced set of features of the policy the average velocity of the robot's pelvis, the desired velocity of the robot, and the error between the desired and the actual average velocity. This selection is inspired by the Hybrid Zero Dynamics (HZD)-based feedback controllers for bipedal locomotion [27] and the simplicity but effectiveness of the LIP model to provide reference trajectories of the COM and step length [28].

C. ACTION SPACE

In our motion planning framework, the action determines the parameterized desired joint trajectories. It has been shown that trajectory actions typically provide a better representation of locomotion than the direct actuator inputs [29]. Parameterized trajectories also allow model-free joint references to be tracked by the feedback controller, thereby enabling the seamless sim-to-real transfer of the learned policy on robot hardware.

As discussed later in this section, the motion policy does not need to determine desired trajectories for all actuated joints of the robot. Let N be the number of actuated joints determined by the motion policy π_y, the desired trajectory of each joint $i \in 0, \dots, N$ will be parameterized as an M-th order Bézier polynomial with coefficients $\alpha_i \in \mathbb{R}^{M+1}$, given as

$$\mathbf{y}_i^d(\tau, \alpha_i) := \sum_{k=0}^{M} \alpha_i[k] \frac{M!}{k!(M-k)!} \tau^k (1-\tau)^{M-k}, \tag{11}$$

where $\tau = \frac{t-t^-}{T_{\text{step}}} \in [0, 1]$ is the scaled time-based phase variable over one walking step with t^- being the time at the beginning of the step, and T_{step} is the time duration of one walking step.

Dimension Reduction of Action Space. In order to reduce the output size, thereby the overall size, of the neural network policy π_y, we reduce the action space dimension by incorporating the unique nature of bipedal locomotion.

Redundant Joints. The desired trajectory of some actuated joints will be directly commanded by the feedback regulator

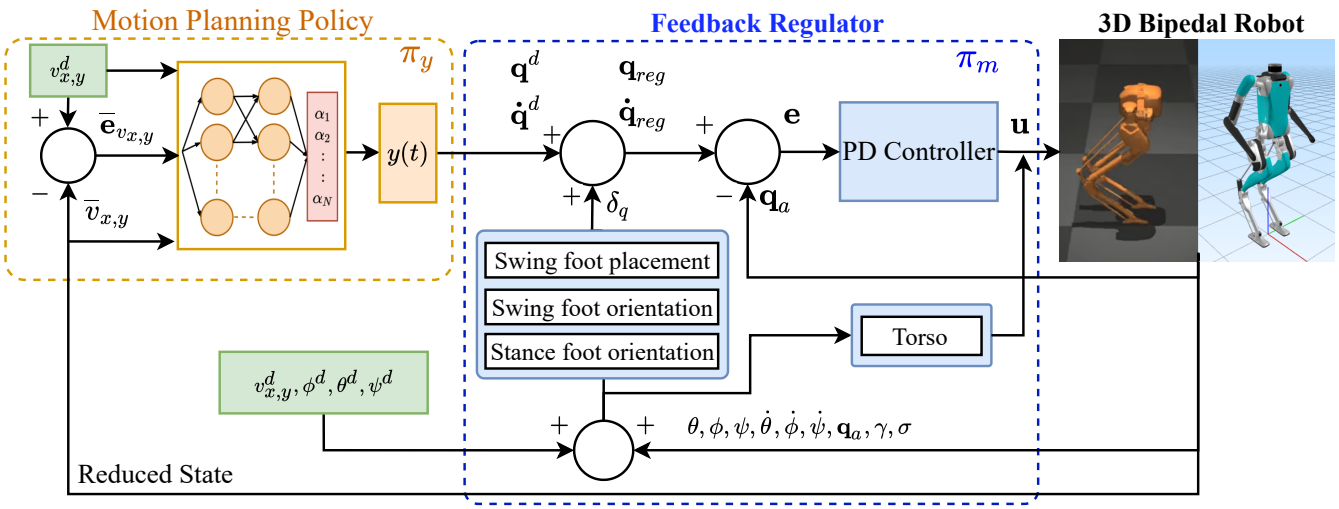

FIGURE 3: Overall structure of the proposed Trajectory-based RL framework. The trajectory planning phase is done by the neural network policy while the feedback regulation block use the robot's sensor information to guarantee the stability of the walking gait and the velocity tracking performance.

policy π_m described in Section IV. Therefore, the motion policy π_y does not need to provide reference trajectories for these joints, significantly reducing the number of outputs required. Specifically, the torso regulation takes care of the stance leg hip roll and pitch joints, the swing foot orientation regulation takes care of the swing ankle roll and pitch joints, and the stance foot regulation takes care of the stance ankle roll and pitch joint. We provided a detailed description of each of these regulations in the following section. Moreover, if arm joints are present (e.g., Digit, see Section V), we can treat the arm as a single pendulum by controlling the motion of the shoulder pitch joint only through the motion policy. Thus, we can lock other arm joints at constant angles, further reducing the policy outputs.

Gait Symmetry. For bipedal locomotion, there exists symmetry between the right and left stance gaits. This allows us to only learn the right stance gait parameters, and determine the left stance gait parameters using the symmetry condition. Assuming that the set of coefficients for the right stance gait α^R is given, the set of coefficients for the left stance gait α^L can be computed by

$$\alpha^L = \mathbf{T}\alpha^R \tag{12}$$

where $\mathbf{T} \in \mathbb{R}^{N\times N}$ is an invertible sparse transformation matrix that captures the symmetry between the robot's joints on the right and left sides.

Impact Invariance. To encourage the smoothness of the control actions after the swing foot impacts the ground, we enforce an equality constraint such that at the beginning of every step, the initial point of the Bézier polynomial (determined by $\alpha_i^R[0]$) coincides with the current position of the i-th robot's joint. To determine the switching condition between right and left stances, we detect the impact of the swing foot with the ground by estimating the ground reaction force (GRF) and comparing it with a fixed threshold easily tuned based on experiments performed both in simulation and hardware. Although this threshold is kept fixed during training and evaluation of the policy, early or late contact conditions are indirectly managed by the learned policy through the update of the reference trajectories at the switching conditions. Finally, we enforce the position of the actuated joints to be the same at the end of the right stance and the beginning of the left stance. This encourages continuity in the joint position trajectories after switching the stance foot. When using Bézier polynomials, this condition can be easily enforced through $\alpha_i^R[M] = \alpha_i^L[0]$. Therefore, two Bézier coefficients for each joint can be obtained through the above conditions. This means we only need to find the remaining $M-1$ coefficients for each of the N reference trajectories, which results in an action space of dimension $N \times M - 1$.

D. LEARNING PROCEDURE

The proposed framework can use any RL algorithm that handles continuous action spaces, including but not limited to evolution strategies (ES), proximal policy optimization (PPO) [2], and deterministic policy gradient (DDPG) [30]. In this work, we use the ES algorithm because of its simple implementation for parallel processing, and its promising results in environments with a high number of time steps in an episode, actions with long-lasting effects, or with no good estimations available for the value function [31]. All of these conditions are present in the problem of bipedal locomotion.

The reward function adopted in this work is determined by a vector of 9 customized rewards with their respective weights $\mathbf{w}$. Specifically:

$$\mathbf{r} = \mathbf{w}^T[r_{v_x}, r_{v_y}, r_{\mathrm{h}}, r_u, r_{\mathrm{CoM}}, r_{\mathrm{ang}}, r_{\mathrm{angvel}}, r_{\mathrm{fd}}, r_{\mathrm{stf}}]^T. \tag{13}$$

These rewards are designed accordingly to the desired prop-

erties described in Section II-B by criteria (3) - (7). That is, encouraging the policy performance in four sub-tasks: velocity tracking, feasible states (height maintenance), admissible actions (energy efficiency), and naturalistic behavior.

To encourage better **velocity tracking** performance for desired average walking speeds in the longitudinal and lateral direction, rewards r_{v_x}, r_{v_y} are defined as

$$r_{v_x} = \begin{cases} \max\left(\rho_v/(\bar{v}_x - v_x^d + \varepsilon), 1\right) & \text{if} \quad |\bar{v}_x - v_x^d| \leq e_{vx} \\ -\rho_v/(\bar{v}_x - v_x^d)^2 & \text{if} \quad |\bar{v}_x - v_x^d| > e_{vx} \end{cases}$$

$$r_{v_y} = \begin{cases} \max\left(\rho_v/(\bar{v}_y - v_y^d + \varepsilon), 1\right) & \text{if} \quad |\bar{v}_y - v_y^d| \leq e_{vy} \\ -\rho_v/(\bar{v}_y - v_y^d)^2 & \text{if} \quad |\bar{v}_y - v_y^d| > e_{vy} \end{cases}$$

where ρ_v is a scaling variable that makes the reward function sharp about the desired walking velocity to encourage better velocity tracking, ε is a bias term to prevent singularities when the tracking error is zero, and e_{vx}, e_{vy} are the bounds for the maximum error allowed in the tracking of the desired average velocity.

To encourage the policy to maintain a desired robot's **height**, we define the reward

$$r_{\text{h}} = \begin{cases} \max\left\{(q_z/q_z^d)^2, 1\right\} & \text{if} \quad |q_z - q_z^d| \leq e_{qz}, q_z \leq q_z^d \\ \max\left\{(q_z^d/q_z)^2, 1\right\} & \text{if} \quad |q_z - q_z^d| \leq e_{qz}, q_z > q_z^d \\ -(q_z - q_z^d)^2 & \text{if} \quad |q_z - q_z^d| > e_{qz} \end{cases}$$

where q_z^d is the desired height and e_{qz} is the maximum error allowed for the height of the robot's base.

The **torque efficiency** reward encourages the learning to reduce the torque applied to the joints.

$$r_u = -\left\|u\right\|^2 \tag{14}$$

Four rewards are designed to encourage the **naturalistic behavior** of the walking gaits by keeping the center of mass inside the support polygon, keeping the torso upright during the walking motion, and keeping the distance between the feet within a desired nominal range. In particular, (15) handles the case when p_{com}^{xy}, the projection of the center of mass on the xy plane, is out of P, the area determined by a radius of $0.1m$ about the midpoint between the projection on the two feet on the xy plane, denoted by Q.

$$r_{\text{CoM}} = \begin{cases} \rho_d/d & \text{if} \quad p_{\text{com}}^{xy} \in P \\ -1/\rho_d(d - 0.1)^2 & \text{if} \quad p_{\text{com}}^{xy} \notin P \end{cases} \tag{15}$$

where ρ_d is a scaling variable, and d is the distance between p_{com}^{xy} and Q.

In (16) and (17), the torso's angles $(q_\psi, q_\theta, q_\phi)$ and angular velocities $(\dot{q}_\psi, \dot{q}_\theta, \dot{q}_\phi)$ are used to penalize the deviation of the torso from an upright position during the walking motion.

$$r_{\text{ang}} = -(q_\psi^2 + q_\theta^2 + q_\phi^2) \tag{16}$$

$$r_{\text{angvel}} = -(\dot{q}_\psi^2 + \dot{q}_\theta^2 + \dot{q}_\phi^2) \tag{17}$$

To prevent that the robot's feet spread apart from each other significantly, or the collision of the feet between each other, a penalization to the reward based on the distance between the robot's feet is added in the form of (18).

$$r_{\text{fd}} = \begin{cases} -(\Delta_f - \Delta_{f\text{min}})^2 & \text{if} \quad \Delta_f < \Delta_{f\text{min}} \\ -(\Delta_f - \Delta_{f\text{max}})^2 & \text{if} \quad \Delta_f > \Delta_{f\text{max}} \\ 0 & \text{otherwise} \end{cases} \tag{18}$$

where $\Delta_{f\text{min}}$ and $\Delta_{f\text{max}}$ are the minimum and maximum desired distance distance between the robot's feet.

Finally, the reward in (19) is used to encourage the stance foot to remain static on the ground.

$$r_{\text{stf}} = -\left\|\mathbf{v}_{\text{stf}}\right\|^2 - \left\|\mathbf{w}_{\text{stf}}\right\|^2, \tag{19}$$

where $\mathbf{v}_{\text{stf}}$ and $\mathbf{w}_{\text{stf}}$ correspond to the linear and angular velocity of the stance foot.

IV. FEEDBACK REGULATOR POLICY DESIGN

The feedback regulator policy π_m modifies some of the trajectories generated by the motion planning policy π_y for some of the robot's joints and generates new trajectories for some other joints. This allows the motion planning policy π_y to reduce the number of outputs needed to be learned, significantly improving the sample efficiency of the learning framework. The regulations applied are intuitive yet powerful and allow the controller to compensate for uncertainties in the model used for training the high-level planner policy and adapt it to unknown disturbances like external forces or challenging irregular terrains that the learned policy has not experienced during training in simulation. These regulations were originally proposed by Raibert in [32], and they have been applied successfully on the control and balance of legged robots in several works, including [33]–[36]. As shown in Figure 3, the feedback regulations are composed of two submodules: i) trajectory regulations and tracking, and ii) direct torque regulations for torso orientation.

A. TRAJECTORY REGULATIONS AND TRACKING

Letting q^d be the desired trajectories for the robot's actuated joints provided by the motion policy π_y, then the regulated trajectories q^{reg} are determined by

$$q^{\text{reg}} = q^d + \mathbf{A}\delta_q, \tag{20}$$

where δ_q is the vector of compensations applied on top of the trajectories for some of the robot's joints directly related with the swing foot placement, swing foot orientation and stance foot orientation. The matrix $\mathbf{A}$ is an assignation matrix that assigns the compensation term with its corresponding joint. Thus, we will use simple PD controllers to track the regulated reference trajectories at the joint level to compute the torque inputs for the actuated joints of the robot. In this paper, the PD controllers are defined as

$$u = -\mathbf{K_p}(q - q^{\text{reg}}) - \mathbf{K_d}(\dot{q} - \dot{q}^{\text{reg}}), \tag{21}$$

where $\mathbf{K_p}$ and $\mathbf{K_d}$ are the matrices of PD gains associated with the actuated joints of the robot.

The following joint regulations are applied in this paper:

$$\delta_q = \left[\delta^{sw}_{hr}, \delta^{sw}_{hp}, \delta^{sw}_{hy}, \delta^{sw}_{tp}, \delta^{sw}_{tr}, \delta^{st}_{tp}, \delta^{st}_{tr}\right]^T, \tag{22}$$

which is determined by:

$$\delta_q = \mathbf{P} \times \mathbf{E} + \mathbf{B}, \tag{23}$$

where, $\mathbf{P}$ is a gain matrix, $\mathbf{E}$ is a vector of velocity errors, and $\mathbf{B}$ is a vector of feed-forward correction terms, respectively defined as follows:

$$\mathbf{P} = \begin{bmatrix} 0 & 0 & S_y K^{sw}_{phr} & S_y K^{sw}_{dhr} & 0 \\ K^{sw}_{php} & K^{sw}_{dhr} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ K^{st}_{ptp} & 0 & 0 & 0 & 0 \\ 0 & 0 & K^{st}_{ptr} & 0 & 0 \end{bmatrix}, \tag{24}$$

$$\mathbf{E} = \begin{bmatrix} \bar{v}_x - v^d_x \\ \bar{v}_x - v^{ls}_x \\ \bar{v}_y - v^d_y \\ \bar{v}_y - v^{ls}_y \\ q_\phi - q^d_\phi \end{bmatrix}, \tag{25}$$

$$\mathbf{B} = \begin{bmatrix} \beta_y & \beta_x & 0 & \xi_{tp} & \xi_{tr} & 0 & 0 \end{bmatrix}^T. \tag{26}$$

The underlying motivation of the joint regulators is described as follows. The **swing foot regulations**, i.e., δ^{sw}_{hp} δ^{sw}_{hr}, and δ^{sw}_{hy}, are originally inspired by the LIP model and has been applied to improve the stability and robustness of model-based feedback controller for 3D bipedal based on the tracking of the average walking speed [32]–[36]. The compensation for lateral speed regulation δ^{sw}_{hr} gives a trajectory compensation for the swing leg's hip roll angle. Analogously, δ^{sw}_{hp}, the compensation for the longitudinal speed regulation, outputs a trajectory compensation for the swing leg's hip pitch joint. Moreover, δ^{sw}_{hy}, the compensation for the heading angle of the robot's torso, adds a trajectory compensation to the swing leg's hip yaw angle to keep the torso's yaw orientation at the desired angle. $S_y \in \{1, -1\}$ depends on the swing foot being left or right, $\bar{v}_x$, $\bar{v}_y$ are the longitudinal and lateral average velocities of the robot, v^{ls}_x, v^{ls}_y are the velocities at the end of the previous step, v^d_x, v^d_y are the reference velocities, and $K^{sw}_{php}, K^{sw}_{dhp}, K^{sw}_{phr}, K^{sw}_{dhr}$ are the proportional and derivative gains of hip pitch and roll joints, respectively. The phase variable τ is used to smooth the regulation at the beginning of each walking step and reduce torque overshoots. The terms β_x and β_y are outputs of an additional PI controller used to compensate for the accumulated error in the velocity and prevent the robot from drifting towards a non-desired direction.

The **swing foot orientation regulations**, i.e., δ^{sw}_{tp} and δ^{sw}_{tr}, are applied to keep the swing foot parallel to the walking surface to ensure a proper landing orientation of the swing foot. These compensations are decoupled for the roll (δ^{sw}_{tr}) and pitch (δ^{sw}_{tp}) joints of the robot's ankle of the swing foot. These regulations are obtained by applying decoupled inverse kinematics (IK) to the robot's leg. Therefore, we represent them as ξ_{tp} and ξ_{tr} in (26) as they are dependant on the kinematic tree of the robot and the slope estimation of the walking surface. To estimate the terrain's slope, we assume the stance foot of the robot is aligned with the terrain's surface, and we use the measurements of the robot's IMU and joint angles to compute the orientation of the stance foot through forward kinematics. In Section V, we provide detailed expressions for these regulations.

Finally, the **stance foot orientation regulations**, i.e., δ^{st}_{tp} and δ^{st}_{tr}, are added to improve the tracking performance of the desired average walking speed. The compensations δ^{st}_{tp} and δ^{st}_{tr} are applied to the stance ankle's pitch and roll joints, respectively, to add a trajectory that modifies the current position of these joints.

B. TORQUE REGULATIONS

The torque regulation module applies torque compensations directly to stance hip joints to maintain the desired torso orientation. The **torso regulation** is used to keep the robot's torso in an upright position, which is desired for a natural motion of the walking gait. Assuming that the stance foot is fixed to the ground during the single support phase and that we have a discrete instantaneous impact during the double support phase, the orientation of the torso is directly controlled by the hip roll and hip pitch joints of the robot's stance leg. Therefore, the PD torque regulation denoted by (u^{st}_{hr}) and (u^{st}_{hp}) can be applied respectively to the hip roll and hip pitch joint of the stance leg to keep the torso upright.

$$\begin{bmatrix} u^{st}_{hr} \\ u^{st}_{hp} \end{bmatrix} = -\begin{bmatrix} K_{p\psi} & K_{d\psi} & 0 & 0 \\ 0 & 0 & S_\theta K_{p\theta} & S_\theta K_{d\theta} \end{bmatrix} \begin{bmatrix} q_\psi - q^d_\psi \\ \dot{q}_\psi - \dot{q}^d_\psi \\ q_\theta - q^d_\theta \\ \dot{q}_\theta - \dot{q}^d_\theta \end{bmatrix}$$

in which, $S_\theta \in \{1, -1\}$ depends on the stance foot being left or right, q^d_ϕ, q^d_θ, $\dot{q}^d_\phi$, and $\dot{q}^d_\theta$ are desired torso roll and pitch angles and angular velocities, and $K_{p\psi}, K_{d\psi}, K_{p\theta}, K_{d\theta}$ are manually tuned PD gains.

V. ILLUSTRATION EXAMPLES

In this section, we present the details of the implementation of the proposed framework on an underactuated bipedal robot Cassie and a humanoid robot Digit, both built by Agility Robotics.

Cassie has 20 degrees of freedom (DoF) and 10 actuated joints. Each leg has five actuated joints corresponding to the motors located on the robot's hip, knee and ankle, and two passive joints corresponding to the robot's shin and tarsus joints. During the single support phase (only one foot on the ground), the robot is underactuated because of its narrow feet.

Digit has the same leg morphology as Cassie, with additional joints for the ankle roll, shoulder, and elbow. This makes Digit a more complex system with 30 DoF and 20 actuated joints. Moreover, Digit is equipped with a full stack

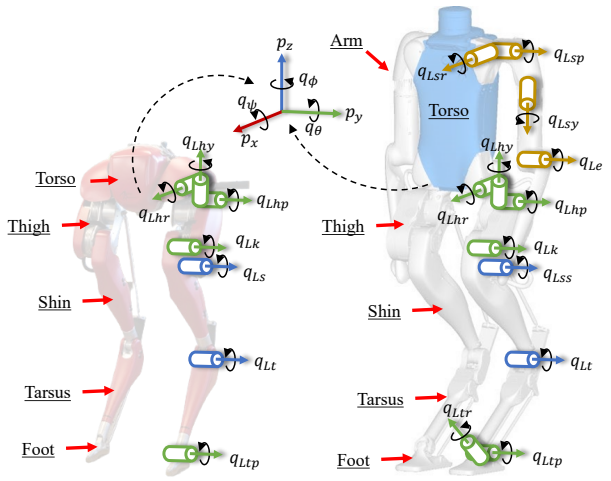

FIGURE 4: Robot model

Description	Cassie	Digit
dim($\mathcal{S}$)	6	6
Outputs of motion policy (N)	6	7
dim($\mathcal{A}$)	24	28
Hidden layers in NN	4	4
Number of units per layer	32	32
Total number of parameters NN	4184	4316

TABLE 1: Details of the state and action space and NN implemented in Cassie and Digit.

of vision sensors, including an RGB camera, four depth cameras, and a LiDAR. Figure 4 shows the kinematic structure of Cassie and Digit with a description of the notation used for the robot's floating base and joints.

A. STATE AND ACTION SPACE

Following the motion policy design presented in Section III-B, the feature state space $\mathcal{S}$ is determined by $s_t = (\eta_t, \zeta_t)$ with $\eta_t = (v_x^d, v_y^d)$, and $\zeta_t = (\bar{v}_x, \bar{v}_y)$, where $(\bar{v}_x, \bar{v}_y)$ are the average longitudinal and lateral velocity, and (v_x^d, v_y^d) correspond to the desired average walking speed. We consider the average speed during one walking step of the robot, which lasts about 400 ms for Cassie and 500 ms for Digit. Similarly, following the considerations discussed in Section III-C, the number of outputs determined for the motion planning policy for Digit is N=7, whereas for Cassie N=6 because we do not have arms motion. More details about the dimension of the state and action spaces are provided in Table 1.

B. NEURAL NETWORK STRUCTURE

The structure of the lightweight neural network used in our framework is shown in Figure 5, and the details about its parameters are shown in Table 1. ReLU activation functions are used between hidden layers, whereas the final layer employs a sigmoid function to limit the range of the outputs. Moreover, Table 2 shows a detailed comparison of the NN structure of our method with state-of-the-art RL frameworks

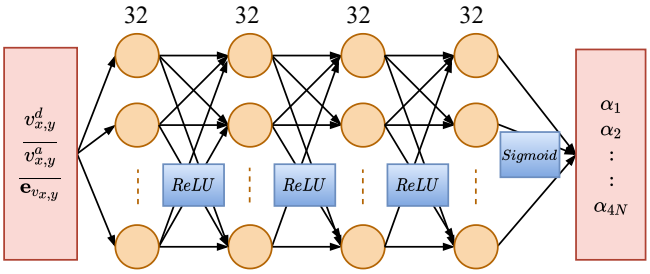

FIGURE 5: Detailed structure of the neural network implemented for the robots Cassie and Digit. By incorporating insights from the symmetry and dynamics of the walking motion, plus simple but effective feedback regulations, we reduce significantly the dimension of the state and action spaces, which results in the smallest NN used for locomotion of real 3D bipedal robots.

Method	State	Action	Layers	Units	Total parameters
Ours	6	24	4	32	4184
[7]	80	10	2	256	89098
[10]	49	10	2	128	225034
[6]	277	10	2	512	410122

TABLE 2: Comparison of the total number of parameters of the neural network with other learning frameworks for bipedal locomotion with the robot Cassie. The neural network implemented in our method has about 20x fewer parameters when compared with the other methods.

for bipedal locomotion. For a fair comparison, we only considered studies implemented on the robot Cassie. Table 2 shows the NN is considerably smaller in size, making the proposed RL framework more lightweighted, faster to train, and feasible to implement on real-time controllers even on budget-limited processors. This is the smallest NN implemented in simulation and hardware to realize robust and stable locomotion on the 3D bipedal robots Cassie and Digit to the best of our knowledge.

C. TRAINING SETUP

To train the NN presented in Section V-B we used the evolution strategies (ES) algorithm [31], using the tuning parameters shown in Table 3. Our learning pipeline uses a model-based balancing controller to obtain a pool of initial

Parameter	Value
Population	24
Standard deviation	0.1
Decay standard deviation	0.9999
Limit standard deviation	1e-4
Learning rate	0.01
Decay learning rate	0.9999
Limit learning rate	1e-4

TABLE 3: Tuning parameters used for training of the policy using Evolution Strategies (ES).

Coefficient	Cassie	Digit
ρ_v	$1e^{-3}$	$1e^{-3}$
ε	$1e^{-5}$	$1e^{-5}$
e_{vx}	0.1	0.1
e_{vy}	0.2	0.2
q_z^d	0.91	1.00
e_{qz}	0.05	0.04
ρ_d	0.01	0.01
d	0.1	0.1
$\Delta_{f\min}$	0.2	0.2
$\Delta_{f\max}$	0.4	0.4
$\mathbf{w}$	$[0.8, 0.2, 0.1, 0.01, 0.1, 0.5, 0.5, 5, 0]^T$	$[0.3, 0.3, 0.2, 0.1, 0.5, 0.5, 0.5, 5, 0.5]^T$

TABLE 4: Coefficients and weights used for the rewards during the training of each environment.

Parameter	Range	Unit
Link Mass	$[0.85, 1.15]$	kg
Link Center of Mass	$[0.95, 1.05]$	m

TABLE 5: Dynamic properties and sample range used for dynamic randomization during training.

states that are feasible to be implemented in both simulation and the real robot. We use a customized environment using MuJoCo [37], with each episode starting from a robot's state chosen randomly from the pool of balanced initial states and uniformly sampled desired walking velocities. We denote that the trained policy learns to walk from scratch without using previously known reference trajectories or policies pre-trained with expert demonstrations. In Table 4, we detail the values of the gains and bounds used for the rewards introduced in Section III-D. We denote that the weight corresponding to r_{stf}, the reward associated with keeping the stance foot static during the step, is equal to zero for Cassie. This reward was added particularly for the Digit because the robot's torso is significantly heavier than Cassie's, which caused Digit's stance foot to slip on the ground. In addition, to encourage policies that realize sustained walking, we increased the episode length from 10000 simulation steps (Cassie) to 15000 (Digit), which are equivalent to 5 and 7.5 seconds, respectively. The episode has an early termination if any of the following conditions are violated:

$$\begin{aligned} &|q_\psi| < 0.5, \quad |q_\theta| < 0.5, \quad |q_\phi| < 0.5, \\ &|\dot{q}_\psi| < 2, \quad |\dot{q}_\theta| < 2, \quad |\dot{q}_\phi| < 2, \\ &0.8 < q_z < 1.2, \quad \Delta_{f\min} < \Delta_f < \Delta_{f\max}, \end{aligned} \tag{27}$$

where q_z is the height of the robot's pelvis and Δ_f is the distance between the feet. In addition, we use dynamic randomization in our training process to improve the robustness of the policy and the sim-to-real transfer success. These parameters are shown in Table 5.

Figure 6 shows the evolution of the normalized mean reward during training for both Cassie and Digit. The number of training episodes needed by the policy to achieve a stable reward is significantly higher in the Digit's environment.

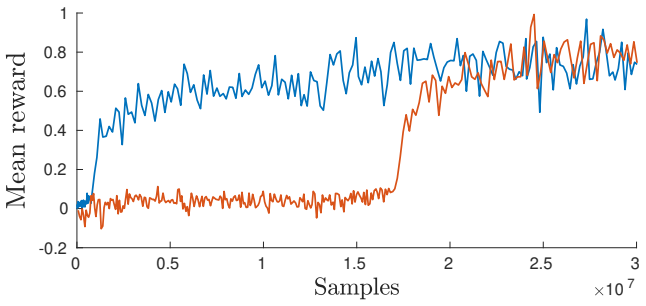

FIGURE 6: Learning process of the trained policy for Cassie (blue) and Digit (red).

	Task	Training time [h]	Samples
Ours	Various speeds	3	0.6e7
[7]	One speed	2.5	-
[10]	Various speeds	8	1e7
[38]	Various speeds	16	-
[6]	Various speeds	-	1e7

TABLE 6: Comparison between different RL frameworks for bipedal locomotion with the robot Cassie.

This result is expected given the higher level of complexity imposed by the model of the Digit robot.

Comparing the sample efficiency between different RL frameworks for bipedal locomotion is difficult because of the particular settings used for each training setup (e.g., learning task, episode length, policy update frequency, learning algorithm, prior knowledge of the walking gait, performance of the trained policy). In addition, not all the methods present information about the number of samples required to learn a stable walking gait. However, Table 6 shows a comparison between state-of-the-art learning-based frameworks for bipedal locomotion. To promote a fair comparison, we only considered methods that use the bipedal robot Cassie. The results show that our method needs fewer samples than other approaches for the reward to converge to a stable value. In addition, Table 6 shows that the proposed framework requires less wall time than other approaches, except for [7], which learns policies that walk at a single desired walking speed using known reference trajectories. On the other hand, our method learns a single policy that tracks various speeds without using known reference trajectories. The policy is trained using a single 12-core CPU machine.

D. FEEDBACK REGULATIONS

The gains of the compensations described in Section IV-A and Section IV-B for Cassie and Digit are detailed in Table 7. We denote that the regulation for the stance foot orientation is applied only to Digit to enhance the speed tracking performance of the controller in hardware experiments. In addition, given the kinematic tree for Cassie and Digit, the IK functions used in the swing foot orientation regulation are defined in Table 7 as ξ_{tr} and ξ_{tp}, where λ_r and λ_p are offsets that depend on the geometric design of the swing leg,

Gain	Cassie	Digit
K^{sw}_{phr}	1	0.5
K^{sw}_{dhr}	0.05	0.03
K^{sw}_{php}	0.6	0.5
K^{sw}_{dhp}	0.01	0.1
K^{st}_{ptp}	-	0.1
K^{st}_{ptr}	-	0.02
$K_{p\psi}$	100	3500
$K_{d\psi}$	20	500
$K_{p\theta}$	100	2000
$K_{d\theta}$	20	500
ξ_{tr}	$q_\psi + q^{sw}_{hr} + \lambda_r + \gamma$	$q_\psi + q^{sw}_{hr} + \lambda_r + \gamma$
ξ_{tp}	$q_\theta + q^{sw}_{hp} + \lambda_p + \sigma$	$q_\theta + q^{sw}_{hp} + \lambda_p + \sigma$

TABLE 7: Gains and IK functions used in the feedback regulator policy for Cassie and Digit.

and γ, σ are the inclination of the terrain with respect to the robot's floating base.

VI. SIMULATION AND EXPERIMENTAL RESULTS

Once the trained policy has been exhaustively tested in simulation, we deploy the learned controller on the hardware and evaluate its performance under challenging conditions and terrains. This section shows the performance of the proposed controller structure when evaluated in terms of speed tracking, stability of the walking gait, and robustness against external disturbances and challenging terrains. A sequence of the learning process of the policy and the sim-to-real transfer can be seen in the accompanying video.

A. SIMULATION RESULTS ON CASSIE

1) Speed Tracking

For evaluation of speed tracking, we assigned a desired velocity profile with fast changes in both longitudinal (v_x) and lateral (v_y) directions with respect to the robot's body frame. The results presented in Figure 7 show that the controller keeps good tracking of the desired velocities in both directions, and it can effectively handle the changes in the speed profile even for large speed changes without significant overshoot. We denote that depending on the combination of the velocity profiles in both directions, the robot can perform different behaviors such as walking in place, walking to the right, left, forward, backward, and walking in a diagonal direction.

2) Stability and feasibility of the walking gait

To evaluate the stability of the generated walking gait, we analyzed the periodicity described by joint limit cycles. Figure 8 shows that the phase portrait for the actuated joints while the robot is walking at a constant desired velocity. The plot shows the convergence of the orbits to a periodic limit cycle, demonstrating the stability of the walking gait. Furthermore, the corresponding orbits for the left and right are approximately symmetrical, which was expected by the conditions enforced in the formulation of the RL framework. The minor discrepancies, mostly noticed in hip roll joints, are due to the swing leg regulator's efforts to maintain the lateral stability of the robot.

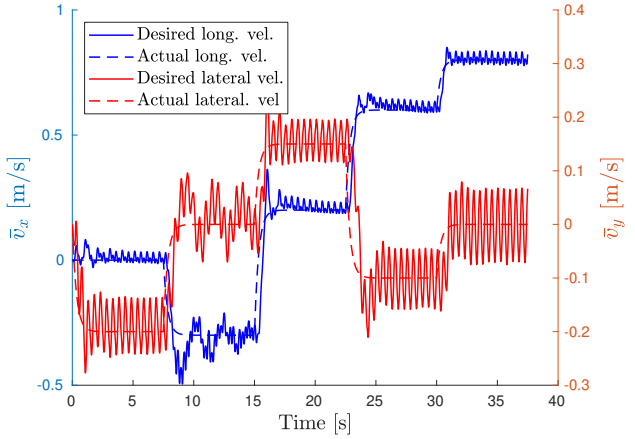

FIGURE 7: Speed tracking performance of the proposed controller in simulation with the robot Cassie. The controller tracks the desired speed for different walking directions: walking forward ($v_x > 0$), backward ($v_x < 0$), to the right ($v_y > 0$), to the left ($v_y < 0$), diagonal (any combination of the previous cases).

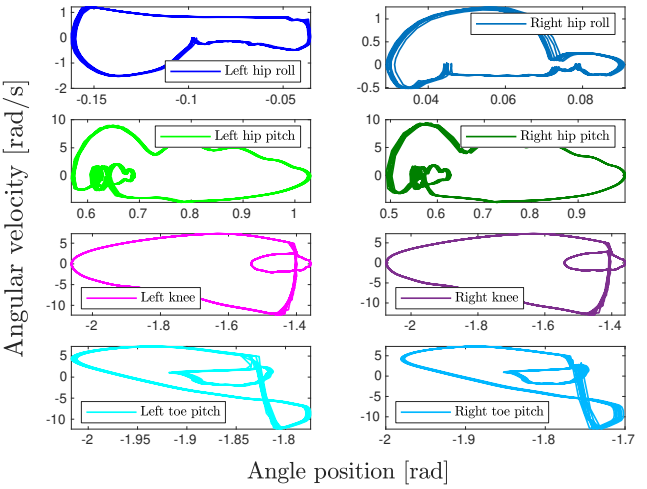

FIGURE 8: Walking limit cycle of the learned policy when tracking a longitudinal velocity $v^d_x = 0.4\ m/s$, and lateral velocity $v^d_y = 0\ m/s$.

B. EXPERIMENTAL RESULTS ON DIGIT

1) Speed Tracking

We evaluate the speed tracking performance of the controller in hardware by assigning a velocity profile with variations in the desired velocities in both directions. The results presented in Figure 9.a show that the controller keeps good tracking of the desired velocities, especially for the velocity in the longitudinal direction ($\bar{v}_x$). We observe that the tracking error is higher for the lateral velocity ($\bar{v}_y$), which could be caused by the continuous motion of the robot from left

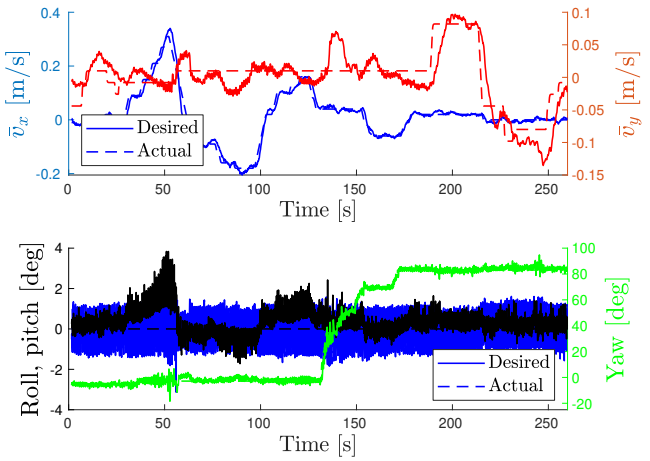

FIGURE 9: Speed tracking performance of the proposed controller. The controller tracks the desired speed for different cases: walking in place ($v_x = 0, v_y = 0$), forward ($v_x > 0$), backward ($v_x < 0$), to the right ($v_y > 0$), to the left ($v_y < 0$), and diagonal (any combination of the previous cases).

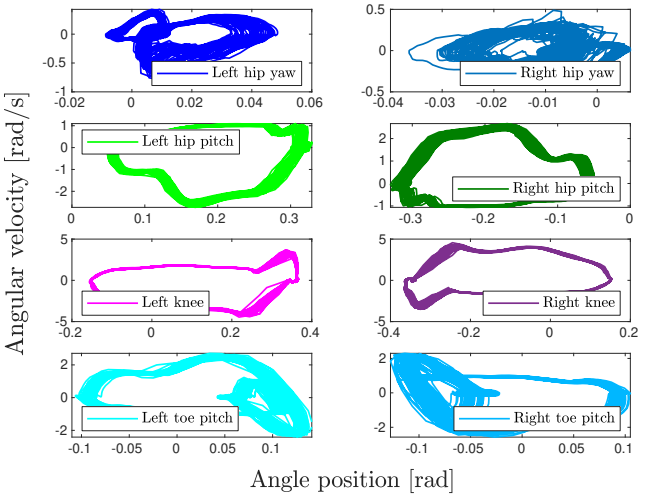

FIGURE 10: Walking limit cycle of the learned policy when tracking a longitudinal velocity $v_x^d = 0.0\ m/s$, and lateral velocity $v_y^d = 0\ m/s$.

to right and vice-versa, asymmetries in the hardware joints associated with the lateral movement, and drifting in the IMU measurements used to estimate the linear velocity. In addition, Figure 9.b shows that the controller keeps the torso upright during the walking gait and accurately tracks the desired heading angle. This tracking performance enables the application of the proposed RL-based cascade motion policy for navigation indoors and outdoors.

2) Stability and feasibility of the walking gait

Figure 10 shows the phase portrait of the actuated joints of the robot's leg while walking at a constant desired velocity. Similar to the simulation results, the plot shows the convergence of the orbits to a periodic limit cycle, empirically demonstrating the stability of the walking gait. As expected, the limit cycles of the joints are noisier than the ones obtained in simulation, particularly for the joints that are being modified by the feedback regulator policy.

3) Robustness

We perform two tests to evaluate the robustness of the cascade controller: i) robustness to external disturbances and ii) robustness when walking on challenging terrain. For the first test, external disturbances are applied to the robot's torso while walking forward ($v_x = 0.11m/s, v_y = 0m/s$). Figure 11 shows the performance of the controller to keep tracking of the desired walking speed, while Figure 12 shows the limit walking cycle of some of the robot joints before, during, and after the disturbance. These results show the controller can recover effectively from disturbances while keeping a good tracking performance of the desired walking speed and maintaining the stability of the walking limit cycle.

For the second test, we set Digit to walk blindly on a

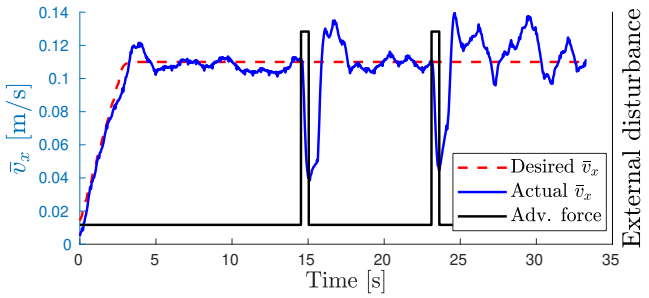

FIGURE 11: Disturbance rejection when external forces are applied in the backward direction

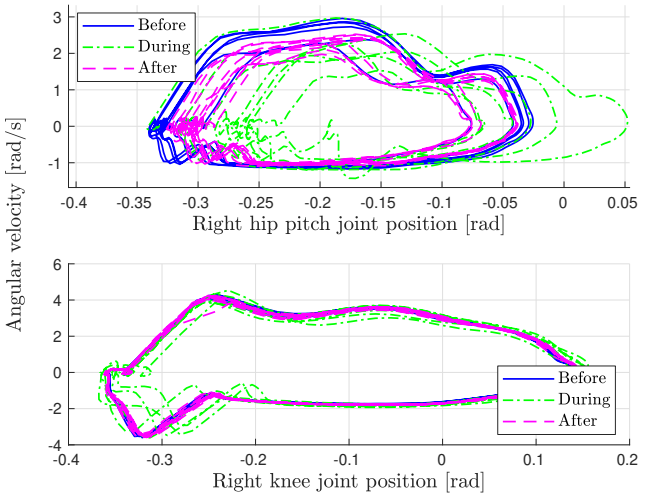

FIGURE 12: Disturbance rejection when adversarial forces are applied in the forward and backward direction

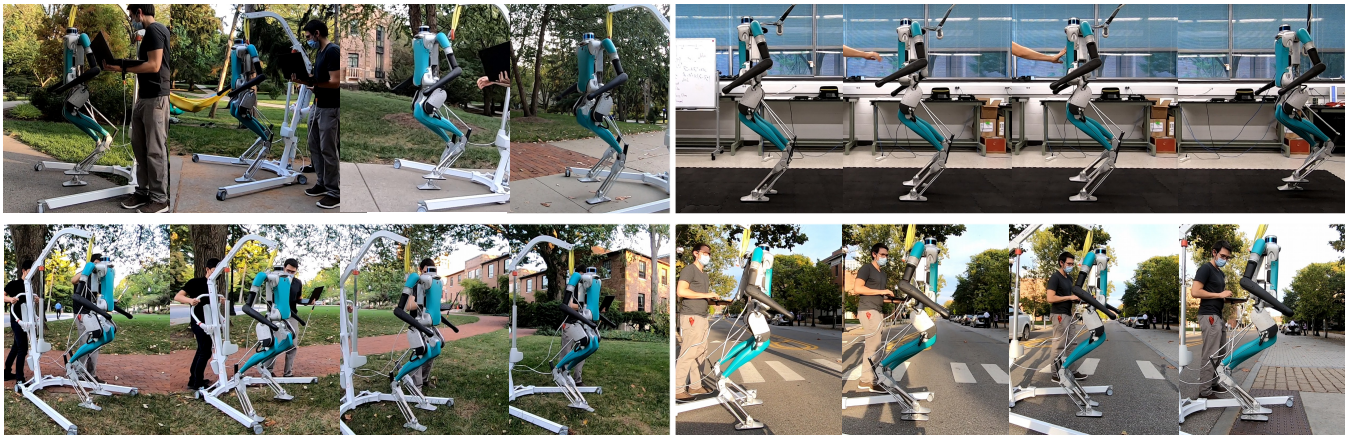

FIGURE 13: Digit walking on different terrains using the learned policy: concrete slopes (top-left), rubber surface (top-right), grass (bottom-left), and pavement (bottom-right). The top-right tile plots also show the robustness of the policy against external forces applied to the torso of the robot.

series of challenging irregular terrains. To test the robustness of the controller on different slopes, we conducted rigorous experiments on a treadmill varying the slope inclination from 0 to 11 degrees. In addition, we evaluate the controller's performance to real-world scenarios by conducting experiments outdoors on different terrains, including concrete ground, vinyl, pavement, grass, and slopes of different inclinations. Figure 13 shows tile plots of the robot walking on some of these terrains. More details about these experiments can be seen in the accompanying video submission.

We evaluate the speed tracking performance of the controller along all the different terrains. The results presented in Figure 14 show the proposed controller structure is able not only to keep stable walking but also to keep a good speed tracking performance on every single terrain. This demonstrates that our learned policy can be used with confidence for navigation in real-world scenarios.

We denote that the same learned policy is used to navigate the robot in all the terrains mentioned above, without the need for additional training or tuning between different terrains. It is important to denote that no disturbances or terrain randomization were applied during the training. Therefore, the robustness of the policy is the result of the enhanced structure of the controller that allows the external and internal loop to be updated at different rates. The inner loop (feedback regulation) facilitates the feedback response of the controller to external disturbances while the outer loop (NN-based trajectory planning) keeps updating the reference trajectories for different desired speeds at a lower rate.

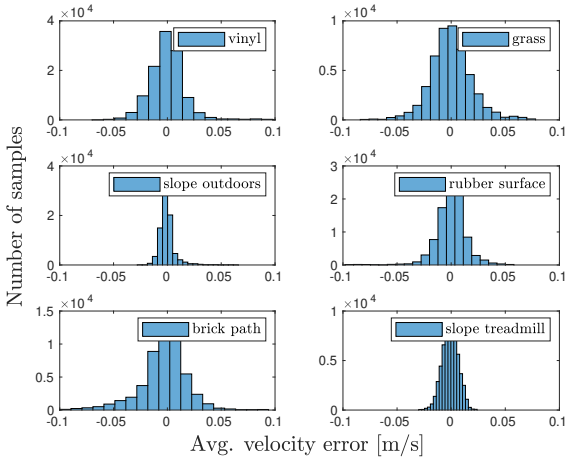

FIGURE 14: Speed tracking performance of the controller while walking blindly on different challenging terrains.

VII. CONCLUSION

This paper presents a novel RL framework for the design of a cascade motion policy that simultaneously addresses two important problems in bipedal locomotion: trajectory planning and feedback regulation. By incorporating the physical insights of dynamic walking such as symmetry motion, invariance through impact condition, and heuristic regulations into the learning process, we provide a complete and effective solution for the design of feedback controllers that realize stable and robust walking gaits without any prior knowledge of reference trajectories. The method relies on a small-size network with reduced state and action spaces, resulting in improved sample efficiency and reduced training time. The proposed method is tested in simulation with two bipedal robots Cassie and Digit, and successful sim-to-real transfer of the learned policy is demonstrated on Digit with minimal tuning. Extensive hardware experiments show the learned policy can track desired walking speeds in any direction while maintaining stable walking gaits. Moreover, the policy is robust to external disturbances and challenging terrains, including rubber ground, pavement, grass, and slopes.

REFERENCES

[1] T. P. Lillicrap, J. J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra, "Continuous control with deep reinforcement learning,"

in *4th Int. Conf. on Learn. Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conf. Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016.

[2] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017. [Online]. Available: http://arxiv.org/abs/1707.06347

[3] R. Tedrake, T. W. Zhang, H. S. Seung *et al.*, "Learning to walk in 20 minutes," 2005, pp. 1939–1412.

[4] J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, and M. Hutter, "Learning agile and dynamic motor skills for legged robots," *Science Robotics*, vol. 4, no. 26, 2019.

[5] T. Haarnoja, S. Ha, A. Zhou, J. Tan, G. Tucker, and S. Levine, "Learning to walk via deep reinforcement learning." in *Robotics: Science and Systems*, 2019. [Online]. Available: https://doi.org/10.15607/RSS.2019.XV.011

[6] Z. Li, X. Cheng, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for robust parameterized locomotion control of bipedal robots," in *2021 IEEE Int. Conf. Robot. Automat. (ICRA)*, 2021, pp. 2811–2817.

[7] Z. Xie, G. Berseth, P. Clary, J. Hurst, and M. van de Panne, "Feedback control for cassie with deep reinforcement learning," in *2018 IEEE/RSJ Int. Conf. Int. Robots Syst. (IROS)*. IEEE, 2018, pp. 1241–1246.

[8] C. Yang, K. Yuan, W. Merkt, T. Komura, S. Vijayakumar, and Z. Li, "Learning whole-body motor skills for humanoids," in *2018 IEEE-RAS 18th Int. Conf. Humanoid Robots*. IEEE, 2018, pp. 270–276.

[9] Z. Xie, H. Y. Ling, N. H. Kim, and M. van de Panne, "Allsteps: Curriculum-driven learning of stepping stone skills," in *Proc. ACM SIGGRAPH/Eurographics Symp. Comput. Animation*. Goslar, DEU: Eurographics Association, 2020. [Online]. Available: https://doi.org/10.1111/cgf.14115

[10] J. Siekmann, S. Valluri, J. Dao, L. Bermillo, H. Duan, A. Fern, and J. Hurst, "Learning memory-based control for human-scale bipedal locomotion," in *Robot. Sci. Syst.*, 2020.

[11] Z. Xie, P. Clary, J. Dao, P. Morais, J. Hurst, and M. van de Panne, "Learning locomotion skills for cassie: Iterative design and sim-to-real," in *Proc. Conf. on Robot Learn.*, L. P. Kaelbling, D. Kragic, and K. Sugiura, Eds. PMLR, 2020, pp. 317–329.

[12] X. B. Peng, P. Abbeel, S. Levine, and M. van de Panne, "Deepmimic: Example-guided deep reinforcement learning of physics-based character skills," *ACM Trans. Graph. (TOG)*, vol. 37, no. 4, pp. 1–14, 2018.

[13] J. Morimoto, G. Cheng, C. G. Atkeson, and G. Zeglin, "A simple reinforcement learning algorithm for biped walking," in *IEEE Int. Conf. Robot. Automat., 2004. Proceedings. ICRA'04. 2004*, vol. 3. IEEE, 2004, pp. 3030–3035.

[14] J. Morimoto, J. Nakanishi, G. Endo, G. Cheng, C. G. Atkeson, and G. Zeglin, "Poincare-map-based reinforcement learning for biped walking," in *Proc. 2005 IEEE Int. Conf. Robot. Automat.* IEEE, 2005, pp. 2381–2386.

[15] X. Da, R. Hartley, and J. W. Grizzle, "Supervised learning for stabilizing underactuated bipedal robot locomotion, with outdoor experiments on the wave field," in *2017 IEEE Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2017, pp. 3476–3483.

[16] T. Li, H. Geyer, C. G. Atkeson, and A. Rai, "Using deep reinforcement learning to learn high-level policies on the atrias biped," in *2019 Int. Conf. Robot. Automat. (ICRA)*. IEEE, 2019, pp. 263–269.

[17] G. A. Castillo, B. Weng, A. Hereid, Z. Wang, and W. Zhang, "Reinforcement learning meets hybrid zero dynamics: A case study for rabbit," in *2019 Int. Conf. Robot. Automat. (ICRA)*, 2019, pp. 284–290.

[18] G. Castillo, B. Weng, W. Zhang, and A. Hereid, "Hybrid zero dynamics inspired feedback control policy design for 3d bipedal locomotion using reinforcement learning," in *IEEE Int. Conf. Robot. Automat. (ICRA)*. Paris, France: IEEE, May 2020.

[19] L. Krishna, U. A. Mishra, G. A. Castillo, A. Hereid, and S. Kolathaya, "Learning linear policies for robust bipedal locomotion on terrains with varying slopes," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. Prague, Czech Republic: IEEE, Dec. 2021.

[20] F. Abdolhosseini, "Learning locomotion: symmetry and torque limit considerations," Ph.D. dissertation, University of British Columbia, 2019.

[21] G.-C. Kang and Y. Lee, "Finite state machine-based motion-free learning of biped walking," *IEEE Access*, vol. 9, pp. 20 662–20 672, 2021.

[22] G. Castillo, B. Weng, T. Steward, W. Zhang, and A. Hereid, "Robust feedback motion policy design using reinforcement learning on a 3d digit bipedal robot," in *Proc. IEEE/RSJ Int. Conf. Intell. Robots Syst. (IROS)*. Prague, Czech Republic: IEEE, Dec. 2021.

[23] Y. Hurmuzlu and D. B. Marghitu, "Rigid body collisions of planar kinematic chains with multiple contact points," *Int. J. Robot. Res.*, vol. 13, no. 1, pp. 82–92, 1994.

[24] X. Da and J. Grizzle, "Combining trajectory optimization, supervised machine learning, and model structure for mitigating the curse of dimensionality in the control of bipedal robots," *Int. J. Robot. Res.*, vol. 38, no. 9, pp. 1063–1097, jul 2019. [Online]. Available: https://doi.org/10.1177/0278364919859425

[25] O. Harib, A. Hereid, A. Agrawal, T. Gurriet, S. Finet, G. Boeris, A. Duburcq, M. E. Mungai, M. Masselin, A. D. Ames, K. Sreenath, and J. Grizzle, "Feedback control of an exoskeleton for paraplegics: toward robustly stable hands-free dynamic walking," *IEEE Control Syst. Mag.*, vol. 38, no. 6, pp. 61–87, Dec. 2018.

[26] R. Chitnis and T. Lozano-Pérez, "Learning compact models for planning with exogenous processes," in *Proc. Conf. on Robot Learn.* PMLR, 2020, pp. 813–822.

[27] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback control of dynamic bipedal robot locomotion*. CRC press Boca Raton, 2007.

[28] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3D linear inverted pendulum model: a simple modeling for a biped walking pattern generation," in *Proc. IEEE/RSJ Int. Conf. Int. Robots Syst.*, 2001, pp. 239–246.

[29] X. B. Peng and h. van de Panne, Michiel, "Learning locomotion skills using DeepRL: does the choice of action space matter?" in *Proc. ACM SIGGRAPH / Eurographics Symp. Computer Animation*, ser. SCA '17. Los Angeles, California: Association for Computing Machinery, Jul. 2017, pp. 1–13.

[30] D. Silver, G. Lever, N. Heess, T. Degris, D. Wierstra, and M. Riedmiller, "Deterministic policy gradient algorithms," in *Int. Conf. on Mach. Learn.* PMLR, 2014, pp. 387–395.

[31] T. Salimans, J. Ho, X. Chen, S. Sidor, and I. Sutskever, "Evolution strategies as a scalable alternative to reinforcement learning," *arXiv preprint arXiv:1703.03864*, 2017.

[32] M. Raibert, H. B. Brown, M. Chepponis, E. Hastings, S. Shreve, and F. C. Wimberly, "Dynamically stable legged locomotion," Carnegie Mellon University, Pittsburgh, PA, Tech. Rep. CMU-RI-TR-81-09, November 1981.

[33] K. Yin, K. Loken, and M. van de Panne, "Simbicon: Simple biped locomotion control," *ACM Trans. Graph.*, vol. 26, no. 3, p. 105–es, jul 2007. [Online]. Available: https://doi.org/10.1145/1276377.1276509

[34] X. Da, O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle, "From 2D design of underactuated bipedal gaits to 3D implementation: Walking with speed tracking," *IEEE Access*, vol. 4, pp. 3469–3478, 2016.

[35] S. Rezazadeh, C. Hubicki, M. Jones, A. Peekema, J. Van Why, A. Abate, and J. Hurst, "Spring-mass walking with ATRIAS in 3D: Robust gait control spanning zero to 4.3 kph on a heavily underactuated bipedal robot," in *ASME 2015 Dyn. Syst. Control Conf.* American Society of Mechanical Engineers, 2015.

[36] Y. Gong, R. Hartley, X. Da, A. Hereid, O. Harib, J.-K. Huang, and J. Grizzle, "Feedback control of a Cassie bipedal robot: walking, standing, and riding a segway," *American Control Conf. (ACC)*, 2019.

[37] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: a physics engine for model-based control," in *2012 IEEE/RSJ Int. Conf. Int. Robots Syst.*, Oct. 2012, pp. 5026–5033.

[38] H. Duan, J. Dao, K. Green, T. Apgar, A. Fern, and J. Hurst, "Learning task space actions for bipedal locomotion," in *2021 IEEE Int. Conf. Robot. Automat. (ICRA)*, 2021, pp. 1276–1282.

GUILLERMO CASTILLO received the electrical engineering degree from Escuela Politécnica Nacional (EPN), Quito, Ecuador, in 2015, and the M.Sc. degree from The Ohio State University, Columbus, USA, in 2019, where he is currently pursuing the Ph.D. degree in electrical and computer engineering.

From 2014 to 2017, he was with the Department of Automation and Control at EPN. His research interests include legged robots, and the combination of reinforcement learning with control systems to develop robust feedback controllers that realize dynamic locomotion.

BOWEN WENG is a Ph.D. student at The Ohio State University, Columbus, USA, where he works on the joint research of nonlinear system, optimal control and reinforcement learning with robotics applications including multi-agent collision avoidance and dynamic locomotion of bipedal robots and humanoids. Mr. Weng received his M.S. in Electrical and Computer Engineering in 2016 from Case Western Reserve University, Cleveland, USA. Mr. Weng is also a technical specialist at Transportation Research Center (TRC) Inc. on assignment to National Highway Traffic Safety Administration (NHTSA), where he has various publications on formal safety verification & validation of Automated Driving Systems (ADS).

WEI ZHANG received a B.E. in Automation from the University of Science and Technology of China in 2003, and a Ph.D. in Electrical Engineering from Purdue University in 2009. From January 2010 to August 2011, he was a Postdoctoral Researcher in the EECS Department at UC Berkeley. He served as an Assistant Professor (2011 – 2017) and then an Associate Professor (with tenure) of Electrical and Computer Engineering at the Ohio State University. In May 2019, he joined the Southern University of Science and Technology (SUSTech), Shenzhen, China, where he is currently a Professor in the SUSTech Institute of Robotics and the Department of Mechanical and Energy Engineering. His research focuses on control and learning theory with an emphasis on applications on robotics and autonomous systems. Dr. Zhang is a recipient of the NSF CAREER award and the Lumley Research Award at the Ohio State University. He served as an Editor of IEEE Transactions on Power Systems between 2015-2017. He is currently a Senior Member of IEEE and an Associate Editor of IEEE Transactions on Control System Technology.

AYONGA HEREID is an Assistant Professor in the Department of Mechanical and Aerospace Engineering at the Ohio State University. He received his B.S. and M.S. degrees in Mechanical Engineering from Zhejiang University in 2007 and 2010, respectively, and received the Ph.D. degree in Mechanical Engineering from the Georgia Institute of Technology in 2016. Prior to joining the Ohio State University, he was a postdoctoral research fellow in the Department of Electrical Engineering and Computer Science at the University of Michigan in Ann Arbor.

His current research interests center around developing advanced nonlinear optimization and control algorithms to realize dynamic and natural locomotion on bipedal robots and lower-limb exoskeletons. He was the recipient of the Best Student Paper Award in 2014 from the ACM International Conference on Hybrid System: Computation and Control and was nominated as the Best Conference Paper Award Finalists in 2016 at the IEEE International Conference on Robotics and Automation.