IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2023, London, UK. Cite as RA-L paper.

Interactive Dynamic Walking: Learning Gait Switching Policies with Generalization Guarantees

Prem Chand, Sushant Veer, and Ioannis Poulakakis

***Abstract*—In this paper, we consider the problem of adapting a dynamically walking bipedal robot to follow a leading co-worker based on physical interaction. Our approach relies on switching among a family of Dynamic Movement Primitives (DMPs) as governed by a supervisor. We train the supervisor to orchestrate the switching among the DMPs in order to adapt to the leader's intentions, which are only *implicitly* available in the form of interaction forces. The primary contribution of our approach is that it furnishes *certificates of generalization* to novel leader intentions for the trained supervisor. This is achieved by leveraging the Probably Approximately Correct (PAC)-Bayes bounds from generalization theory. We demonstrate the efficacy of our approach by training a neural-network supervisor to adapt the gait of a dynamically walking biped to a leading collaborator whose intended trajectory is not known explicitly.**

***Index Terms*—Humanoid and Bipedal Locomotion, Legged Robots, Gait Switching, PAC-Bayes Generalization**

I. Introduction

IMAGINE a bipedal robot physically assisting a human to carry an object. The human knows where and how the object needs to be placed and the spatial layout of the area. Based on this information, a plan of action can be quickly devised by the human. More often than not, this plan cannot be explicitly—e.g., as a desired trajectory—communicated to the robot. Yet, in physically coupled dyads, the forces developed at the port of interaction between the robot and the human encode important information regarding the human's action plan. This paper aims at enabling bipedal robots to make decisions as to how to *interact* and modify their gaits to follow intended—yet *unknown*—trajectories.

Engaging bipedal robots in tasks that require physical interaction with a collaborator has received considerable attention in the relevant literature; see [1] for a comprehensive collection of related methods. Focusing on the implications of physical interaction to gait adaptation, the vast majority of these efforts—see [2]–[4] and references therein for examples—involve humanoid robots that walk under the Zero Moment Point (ZMP) stability criterion, suitably combined with *reactive* walking pattern generators to ensure that the robot safely adapts to the collaborator's intentions. By way of contrast, gait adaptation for physical interaction has not been explored in dynamically walking[1] bipeds. Indeed, much of the relevant research focuses on feedback control design methods [5]–[7], with recent results addressing motion planning [8]–[10] and trajectory tracking [11] under the assumption that the plan and the desired trajectory are known. Recently, [12] realized walking without reference trajectories by employing domain randomization during training to learn robust policies for traversing uneven terrain without exteroceptive sensors.

Typically, many of these approaches treat externally applied forces not related to locomotion as disturbances to be attenuated. A different approach has been adopted in [13], [14], in which adaptation of periodic walking gaits to exogenous forces has been considered within the Hybrid Zero Dynamics (HZD) framework [5]. To extend the span of influence of these controllers, a supervisory switching control approach was introduced in [15] and stability issues associated with gait composition were investigated in [16], [17]. The purpose of the present paper is to extend this approach and improve the ability of a dynamically walking biped to track an intended (unknown) trajectory. This will be achieved by suitably interpreting interaction forces and making decisions as to which out of a library of controllers must be engaged at each stride.

Being able to adapt to intentions communicated indirectly via interaction forces is a central problem in physical human-robot interaction (pHRI). In this context, a significant body of work focuses on admittance control schemes to "translate" externally applied forces to reference trajectories [18]. Although a detailed account of the relevant literature would take us too far afield, we note here that the choice of the parameters that govern the dynamic relation between the force and the desired motion is critical [19]. This observation motivated the development of various variable admittance control schemes [19]–[21], which typically assume a finite number of intention states; e.g., accelerate, decelerate, stop. To address more general situations, [22] proposed learning a task model via demonstration; this model was then used to predict the perceived force and generate reference trajectories accordingly. Another learning-based approach was discussed in [23], where the collaborator's intention is represented by a desired trajectory that is *not* explicitly known and is estimated via a neural network (NN). Finally, [24] introduced an inner/outer-loop architecture, wherein the outer loop is an adaptive admittance scheme that uses intent estimation to convert applied forces to desired positions.

To bring dynamic walkers a step closer to executing tasks that involve physical interaction, this paper adopts a hierarchi-

Manuscript received: September, 23, 2021; Revised December, 18, 2021; Accepted January, 13, 2022.

This paper was recommended for publication by Editor Dr. Abderrahmane Kheddar upon evaluation of the Associate Editor and Reviewers' comments. This work was supported in part by NSF CAREER Award IIS-1350721.

P. Chand and I. Poulakakis are with the Department of Mechanical Engineering, University of Delaware, Newark, DE, USA; e-mail: {premc, poulakas}@udel.edu. S. Veer is with NVIDIA Research, USA; email: sveer@nvidia.com. This work was conducted while S. Veer was with the Department of Mechanical and Aerospace Engineering, Princeton University, Princeton, NJ, USA.

Digital Object Identifier (DOI): see top of this page.

[1]Here, the term "dynamically walking" refers to limit-cycle walkers [5].

IEEE Robotics and Automation Letters (RA-L) paper, presented at ICRA 2023, London, UK. Cite as RA-L paper.

cally structured approach that relies on learning dynamic gait switching policies in response to externally applied forces; these forces are assumed to encode intended, yet unknown, desired trajectories. Unlike previously proposed methods, our learning algorithm is accompanied with provable performance guarantees on novel collaborator intentions; that is, *generalization guarantees*. Generalization refers to the ability of a learned function to perform well on test sets that are different from the training sets, albeit drawn from the same distribution. Our approach falls within the purview of the Probably Approximately Correct (PAC)-Bayes framework, which recently demonstrated the ability to provide strong generalization guarantees on deep neural networks [25]. Harnessing the recently developed PAC-Bayes control framework [26], [27], the dynamic gait switching policies learnt in this paper are accompanied with explicit bounds on performance under novel leader intentions, as these are communicated via interaction forces. Beyond interactive dynamic walkers, we believe that bounds of this sort are important in the context of pHRI, since they essentially quantify the *risk* of applying control policies to robots engaged in tasks that involve physical contact.

II. Overview of the Method and the Task

This section provides an overview of the proposed approach and describes the main components in general terms.

A. Method: Adaptation via Learning Gait Switching

Building on our recent results [15]–[17], we formulate the problem of adapting dynamic locomotion to externally applied forces as a switching system among a collection of dynamic walking gait controllers regulated by a high-level supervisor; see Fig. 1. In the absence of external forces, each of these controllers generates local dynamics with equilibrium solutions corresponding to periodic walking motions—i.e., *attractor landscapes*—giving rise to *Dynamic Movement Primitives* (DMPs) [28]. The execution of a DMP results in a displacement of the robot in its workspace, as shown in Fig. 1. To decide which controller must be engaged, the supervisor incorporates a NN that learns—with probabilistic generalization guarantees—how to interpret noisy information about the interaction forces as a directional indicator associated with an unknown intended trajectory. The low-level feedback control loop then executes the controller—equivalently, the DMP—suggested by the supervisor.

The hierarchical structure of the proposed approach effectively *decouples* locomotion stability from adaptation. Locomotion stability in the presence of external forcing can be studied in a tractable fashion by applying the theoretical tools developed in [16], [17], [29]; we defer to these references for more details. This paper focuses on adaptation through learning with generalization guarantees. In particular, we train the NN supervisor by minimizing the PAC-Bayes generalization bound, which results in a "certificate of performance" on novel leader trajectories. Note here that adaptation is not part of the low-level control design, which focuses on realizing stable locomotion; instead, adaptation occurs at the high level, by selecting the controller which is more suitable based on stride-to-stride (low-frequency) feedback to the supervisor. An important consequence of this hierarchy is *modularity*. Although the supervisor's ability to adapt the system's behavior relies on properties of the individual controllers, it does *not* rely on the specific low-level design details of how these properties are realized. Thus, the proposed framework is not tied to a particular control design method, as long as switching among the low-level controllers is safe—in the sense defined in [17], [29]—and can generate sufficiently rich locomotion behaviors.

B. Task Description: Intention via Interaction

In this paper, we apply the proposed learning and control hierarchy to realize dynamic gait adaptation based on interaction forces. In more detail, we will assume that the leader's intention over a time interval $[0, T]$ with $T > 0$ is encoded in a sufficiently smooth desired trajectory $\mathrm{p}_{\mathrm{L}}(t)$ where $t \in [0, T]$.

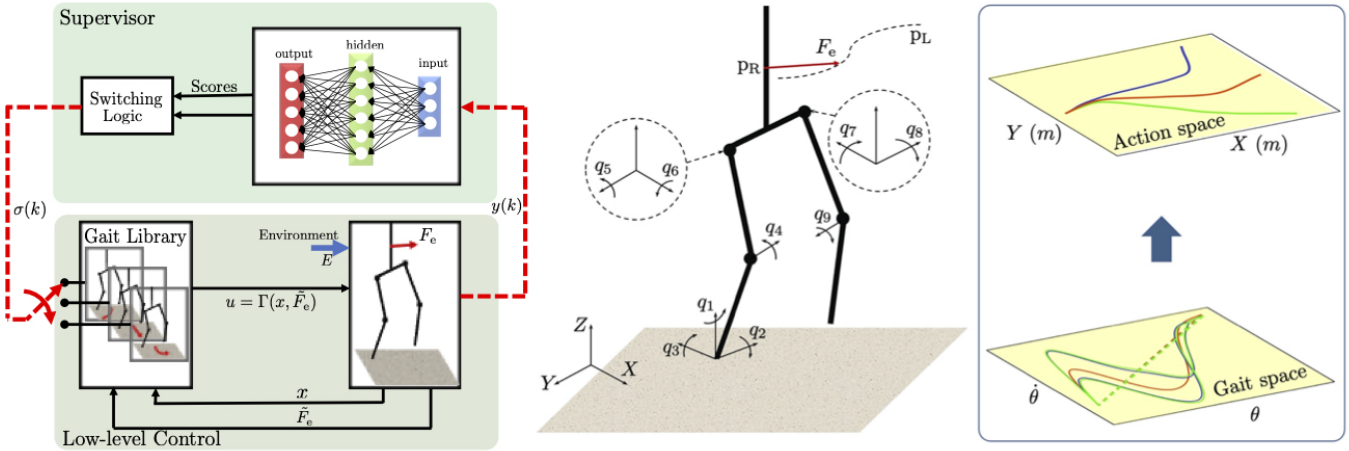

Fig. 1: (Left) Overview of the proposed framework. The high-level supervisor processes stride-to-stride observations $y(k)$ about the robot's motion and interaction forces, and returns the index $\sigma(k)$ of the gait controller that must be engaged during the k-th step. The supervisor relies on a NN that learns—with provable generalization guarantees—how to interpret interaction force measurements as a directional indicator of the unknown desired trajectory. The low-level feedback loop executes the suggested gait controller. (Middle) A 3D bipedal robot model with a choice of coordinates. The biped experiences an interaction force $F_{\mathrm{e}}(t)$ that carries information about a desired, yet unknown, trajectory $\mathrm{p}_{\mathrm{L}}(t)$. (Right) On the low level, a gait controller design method is employed to extract a library of limit-cycle gait primitives. On the high level, each gait primitive generates an action that represents suitable displacements.

The biped does *not* explicitly know $\mathrm{p_L}(t)$; instead, it "perceives" the intended trajectory via a 3D force vector $F_\mathrm{e}(t)$ developed at the port of interaction between the leader and the biped. We will assume that the biped and its leading co-worker collaborate in a constructive way so that the applied force is consistent with the desired trajectory of the leader. We will also assume that noisy measurements $\tilde{F}_\mathrm{e}(t)$ of $F_\mathrm{e}(t)$ are available via a force sensor. To simplify the exposition without changing the essential features of the problem, we will consider the case where the port of interaction is a point $\mathrm{p_R}$ on the robot's torso, as shown in Fig. 1 (middle). Note that in the case where the biped interacts via an arm, $\mathrm{p_R}$ would correspond to the position of the end effector; see [14].

III. PAC-Bayes Switching Policies for Adaptation

This section formalizes the main ideas of the proposed framework; see Fig. 1. Subsequent sections particularize these ideas and discuss the relevant computational tools.

A. From Limit-cycle Gaits to Dynamic Movement Primitives

Dynamic bipedal walking can be modeled by distinguished periodic solutions—that is, *limit cycles*—of hybrid robot models [5]. Consider a collection of limit cycles $\{\mathcal{O}_r \mid r \in \mathcal{R}\}$, where $\mathcal{R}$ is a finite index set. The limit cycles $\mathcal{O}_r$ are designed to capture walking gaits with different attributes; e.g., walking straight, turning with different angles, or other behaviors relevant to the task. Poincaré's method [5] associates each limit cycle $\mathcal{O}_r$ with an equilibrium (fixed) point of a discrete dynamical system that captures the stride-to-stride evolution of the biped. Let $\mathcal{X} \subset \mathbb{R}^n$ denote the state space of the robot and let $\mathcal{S} \subset \mathcal{X}$ be a surface transversal to $\mathcal{O}_r$ for all $r \in \mathcal{R}$; typically, $\mathcal{S}$ is selected to be the ground surface. Then, $\mathcal{O}_r$ can be represented by a fixed point $x_r^* \in \mathcal{S}$ of the system

$$x_{k+1} = f_r^*(x_k) \ , \qquad r \in \mathcal{R} \tag{1}$$

where $x \in \mathcal{S}$ denotes the state, $f_r^* : \mathcal{S} \to \mathcal{S}$ is the corresponding Poincaré map and $x_r^* = f_r^*(x_r^*)$; see [5].

With this construction, the behavior of the system *locally* around a limit-cycle walking gait $\mathcal{O}_r$ can be formalized as a 2-tuple containing the map f_r^* and its fixed point x_r^*; i.e. $\mathcal{G}_r = \{f_r^*,\ x_r^*\}$ with $r \in \mathcal{R}$. Thus, each $\mathcal{G}_r$ defines a Dynamic Movement Primitive (DMP) and the collection $\mathbb{G} = \{\mathcal{G}_r \mid r \in \mathcal{R}\}$ forms the library of DMPs available to the supervisor [16].

B. The Supervisor: Adapting via Switching

The supervisor processes incoming information about the robot and its environment, and decides which DMP must be implemented at the ensuing stride. Here, the term "environment" encompasses effects that are external to the robot. For example, an unknown trajectory $\mathrm{p_L}(t)$ representing the leading co-worker's intent will be considered as part of the environment. Other effects, such as random initial conditions, noisy measurements, model uncertainty, or workspace geometry can also be considered as parts of the environment. To emphasize the role of an environment E on the robot's motion when primitive $\mathcal{G}_r$ is executed, we write

$$x_{k+1} = f_r(x_k;\ E) \ , \qquad r \in \mathcal{R} \tag{2}$$

where E is assumed to belong in some set $\mathcal{E}$ of environments and $f_r : \mathcal{S} \times \mathcal{E} \to \mathcal{S}$ is the state update rule. Comparing (2) with (1), we can interpret (1) as the evolution of the system in a *nominal* environment $E^* \in \mathcal{E}$ for which the gait primitives are derived; i.e., $f_r^*(x) = f_r(x;\ E^*)$. However, different environments are encountered during task execution, causing the state to evolve according to (2).

The information available to the supervisor can be captured by a mapping $H : \mathcal{X} \times \mathcal{E} \to \mathcal{Y}$ that furnishes a partial observation, i.e., *cue*, $y \in \mathcal{Y}$ from a state $x \in \mathcal{X}$ and an environment $E \in \mathcal{E}$. In the context of a biped following an unknown trajectory, such cues may include gait features—e.g., heading or speed—as well as measurements of the interaction forces developed during the task. Based on this information, the supervisor assigns the index $r = \sigma(k)$ of the gait $\mathcal{G}_r \in \mathbb{G}$ that is required at each stride k. This gives rise to the (perturbed) switched system with multiple equilibria [17],

$$x_{k+1} = f_{\sigma(k)}(x_k;\ E) \tag{3}$$

which describes the dynamics of the robot in response to the supervisor's sequence of decisions. Note that (3) describes a system that "sways" among fixed points. Persistent switching causes (3) to be in a "perpetual" transient phase, never converging to any of the underlying fixed points. More information on how to define safety for such systems can be found in [16], [17]; there, *explicit* guarantees for safe operation in the sense of practical stability are provided.

C. Learning Provably Generalizable Switching Policies

In the context of adapting to intention via interaction, we are concerned with learning policies $\pi : \mathcal{Y} \to \mathcal{R}$ in a policy space Π that map "cues" $y \in \mathcal{Y}$ regarding the state of the system and its environment to the index $r \in \mathcal{R}$ of the DMP $\mathcal{G}_r$ in $\mathbb{G}$ that is "best" to employ. Importantly, our goal in this work is to learn switching policies, which, given a dataset of environment instances, *generalize with provable guarantees* to novel environments. To achieve this, we will utilize PAC-Bayes theory, which is known to provide strong generalization bounds in supervised learning [30], [31].

We begin by assuming the availability of a cost function $C : \Pi \times \mathcal{E} \to [0, 1]$, which captures critical aspects of the task and can be used to assess the "quality" of employing a policy $\pi \in \Pi$ in a particular environment $E \in \mathcal{E}$. Note that there is no loss of generality in constraining the co-domain of the cost function to $[0, 1]$; indeed, any bounded cost function could be used as long as it is scaled with a suitable constant.

Next, we assume that there is a distribution $\mathcal{D}$ over the space $\mathcal{E}$ of possible environments; this distribution reflects the underlying stochastic mechanism by which an environment is encountered by the system. It is important to emphasize that we do *not* assume knowledge of $\mathcal{D}$. In this setting, our objective is to learn policies that minimize the expected cost across environments generated according to $\mathcal{D}$,

$$\min_{\pi \in \Pi} C_\mathcal{D}(\pi) = \min_{\pi \in \Pi} \mathop{\mathbb{E}}_{E \sim \mathcal{D}} [C(\pi;\ E)] \ . \tag{4}$$

To formulate the optimization problem (4) so that PAC-Bayes theory can be applied, we will randomize the policy

space Π. This is done by assuming a distribution P over Π according to which individual policies can be selected. In this setting, when the robot encounters an environment E, the supervisor randomly selects a policy from P and applies it to decide which gait primitive should be engaged in the forthcoming stride. With this modification, if $\mathcal{P}$ denotes the space of distributions defined over Π, our goal becomes to learn policy distributions $P \in \mathcal{P}$ that realize the minimum

$$C^\star = \min_{P\in\mathcal{P}} C_{\mathcal{D}}(P) = \min_{P\in\mathcal{P}} \mathop{\mathbb{E}}_{E\sim\mathcal{D}} \left[\mathop{\mathbb{E}}_{\pi\sim P}[C(\pi; E)] \right] \ . \tag{5}$$

However, as was mentioned above, the distribution $\mathcal{D}$ is not known, and thus the expectation over $\mathcal{D}$ in (5) cannot be explicitly computed. Yet, *indirect* knowledge about $\mathcal{D}$ can be obtained by sampling the space of environments $\mathcal{E}$, resulting in datasets corresponding to (finite) collections of environments $D = \{E_1, E_2, ..., E_N\}$. Then, the expectation over $\mathcal{D}$ can be approximated by the empirical average

$$C_D(P) = \frac{1}{N} \sum_{E\in D} \mathop{\mathbb{E}}_{\pi\sim P}[C(\pi; E)] \ . \tag{6}$$

The PAC-Bayes generalization framework [25], [30] provides a computable upper bound on the expected true cost $C_{\mathcal{D}}(P)$ in (5) in terms of the empirical cost $C_D(P)$ in (6).

To apply the PAC-Bayes framework, we assume the availability of a "prior" distribution $P_0 \in \mathcal{P}$ *before* observing any data. Note that P_0 is *not* a Bayesian prior; that is, the correctness of the PAC-Bayes bound is not subject to the correctness of the prior, thus providing enhanced flexibility in the choice of P_0. The crucial benefit of this flexibility is that (partial) knowledge of the problem can be embedded as inductive bias in the learning framework without compromising the correctness of the bounds. Theorem 1 provides an expression for the PAC-Bayes upper bound on the true cost $C_{\mathcal{D}}(P)$ that does *not* rely on knowing $\mathcal{D}$ explicitly, and can therefore be computed using data samples; for a proof see [27].

Theorem 1 (adapted from [27], [25]): Let $\delta \in (0, 1)$, $D \sim \mathcal{D}^N$ be a multisample $D = \{E_1, E_2, ..., E_N\}$ of N training environments drawn in an independent and identically distributed (i.i.d.) fashion from $\mathcal{E}$ according to $\mathcal{D}$, and $P_0 \in \mathcal{P}$ be a prior distribution on the space of policies Π. Then, with probability at least $1-\delta$, for any posterior distribution $P \in \mathcal{P}$, the following inequality holds:

$$C_{\mathcal{D}}(P) \le C_{\mathrm{QPAC}}(P, P_0) := \left(\sqrt{C_D(P) + R(P, P_0)} + \sqrt{R(P, P_0)} \right)^2 \tag{7}$$

in which $C_D(P)$ is given by (6) and $R(P, P_0)$ is defined as

$$R(P, P_0) = \frac{\mathrm{KL}(P||P_0) + \log(\frac{2\sqrt{N}}{\delta})}{2N} \tag{8}$$

where $\mathrm{KL}(P||P_0)$ denotes the Kullback-Leibler divergence (relative entropy) from P_0 to P.

The importance of Theorem 1 is that we can find a posterior policy distribution P by minimizing the bound C_{QPAC}, which consists of two terms: (i) the empirical cost $C_D(P)$, and (ii) the regularizer $R(P, P_0)$. Intuitively, minimizing $C_D(P)$ tries to "fit" the posterior P to the training data D, while the regularizer R penalizes over-fitting. Then, equipped with a prior P_0 and samples D of environments from $\mathcal{E}$, finding a posterior distribution that minimizes the PAC-Bayes bound (7) can be done in a computationally tractable manner using convex optimization tools; see Section V-C.

Remark 1: Any approach can be employed to obtain a suitable prior distribution P_0; the only criterion that must be satisfied is the independence of P_0 from the training dataset D that is used for the application of Theorem 1. The flexibility in choosing P_0 allows us to design informative priors by leveraging highly parallelizable Evolutionary Strategies (ES), as described in Section V-C below.

IV. Low-Level Control and Gait Library

Here, we describe the biped model and the feedback controller design approach used to extract the gait library.

A. Robot Model and Controller Design

One advantage of our hierarchical approach is that the library of low-level controllers can be extracted using *existing* locomotion control design methods without the need of major modifications. To demonstrate this point, we adopt here the 3D bipedal robot model of Fig. 1 (middle), for which walking controllers are available in the relevant literature [14], [32]. The parameters of this model can be found in [32, Table I]. Note though that other dynamically walking robot models and control design methods can also be used.

In our setting, bipedal walking is characterized by a sequence of alternating left and right leg support phases punctuated by instantaneous double support phases [32]. In single support, the model possesses nine degrees of freedom (DOF), which can be described by the coordinates $q = (q_1, q_2, ..., q_9)$; see Fig. 1. In designing a controller for the single support phase, we assume that all DOFs except from the yaw q_1 and pitch q_2 angles of the foot are actuated[2].

Due to the non-trivial length of the hip, the equations of motion for the left and right (single) support phases differ, but both can be written as systems with impulse effects,

$$\mathcal{H} : \begin{cases} \dot{x} = \alpha(x) + g(x)u + g_{\mathrm{e}}(x)F_{\mathrm{e}}, & x \notin \mathcal{S} \\ x^+ = \Delta(x^-), & x^- \in \mathcal{S} \end{cases} \tag{9}$$

where $x = [q^{\mathsf{T}}, \dot{q}^{\mathsf{T}}]^{\mathsf{T}}$ and the vector fields $(\alpha, g, g_{\mathrm{e}})$ describe the single support dynamics under the input u and the interaction force F_{e}. In (9), $\mathcal{S}$ is the ground surface and Δ maps the state x^- prior to impact to the state x^+ after impact.

B. Controller design and gait primitives

Here, we turn our attention to the design of the low-level controllers for walking. We will adopt the HZD framework [5] with virtual (holonomic) constraints designed as in [32], [33]; thus, our exposition will be terse. To the continuous part of (9), we associate the output function

$$y = h(q) = q_{\mathrm{a}} - h_{\mathrm{d}}(\theta(q)) - h_{\mathrm{c}}(\theta(q)) - h_{\mathrm{s}}(\theta(q), \beta) \tag{10}$$

[2]Implicit is the assumption that the robot's feet have nontrivial dimensions and that their mass is small enough to be considered negligible.

where $q_{\rm a}$ includes the actuated variables and $\theta(q) = -q_2 - q_4/2$ is the angle of the line connecting the hip and the foot of the support leg. The terms $h_{\rm d}$ and $h_{\rm c}$ are designed according to [33] to realize straight-line periodic walking gaits in the *absence* of the external force. Given a straight-line walking gait, the term $h_{\rm s}$ is designed next as in [32] to induce turning. Note that $h_{\rm s}$ has the property $h_{\rm s}(\theta, 0) = 0$ so that straight-line walking corresponds to $\beta = 0$.

A discrete collection of values $\{\beta_r,\ r \in \mathcal{R}\}$ is then selected to obtain periodic turning motions with different radii, thus resulting to the set of outputs $\{h_r,\ r \in \mathcal{R}\}$. In the absence of external forcing, these outputs are imposed on the continuous-time part of (9) using standard output-zeroing controllers [5]; these controllers in closed loop with (9) generate the library $\mathbb{G}$ of DMPs $\mathcal{G}_r = \{f_r^*,\ x_r^*\}$. In this paper, $\mathbb{G}$ contains periodic walking gaits with turning angles in the range $[-45°, 45°]$ with $5°$ increments; see Fig. 2 (left).

During interaction, the output-zeroing controllers are slightly modified to account for the external force. In more detail, differentiating the outputs $\{h_r,\ r \in \mathcal{R}\}$ designed above along the continuous-time part of (9) results in

$$\ddot{y} = L_\alpha^2 h(x) + L_g L_\alpha h(q) u + L_{g_{\rm e}} L_\alpha h(q) F_{\rm e}\ .$$

Assuming that $L_g L_\alpha h(q)$ is invertible and that feedback of the force $F_{\rm e}$ is available, a controller $u = \Gamma_r(x, F_{\rm e})$ is designed as in [13, Section II-B]. Closing the loop of (9) with Γ_r results in a *forced* system with impulse effects [29]. Although a rigorous stability analysis for such systems is beyond the scope of this work, it was shown in [29] that if the fixed point x_r^* of the unforced stride map f_r^* is (locally) exponentially stable—which can be checked by computing the eigenvalues of the linearization of f_r^* at x_r^*—then the corresponding limit cycle $\mathcal{O}_r$ is (locally) input-to-state stable. Thus, assuming that the biped and the leader collaborate constructively, the interaction forces will not be excessive and the evolution of the system will remain bounded.

V. Learning to Adapt via Physical Interaction

This section provides details on training switching policies that provably generalize to novel environments.

A. Environment Generation

Over the interval $[0,\ T]$, the evolution of the closed-loop system depends on the initial condition $x_0 = x(0)$ and on the unknown desired trajectory $\{{\rm p_L}(t) \mid t \in [0, T]\}$, which are considered as part of the environment of the system. To "translate" the intended trajectory ${\rm p_L}(t)$ to an interaction force, we adopt a compliance model, according to which

$$F_{\rm e}(t) = K_{\rm L}\Big({\rm p_L}(t) - {\rm p_R}(t)\Big) + N_{\rm L}\Big(\dot{\rm p}_{\rm L}(t) - \dot{\rm p}_{\rm R}(t)\Big) \quad (11)$$

where $K_{\rm L}$ and $N_{\rm L}$ are 3×3 stiffness and damping matrices, respectively, and ${\rm p_R}$ denotes the point on the robot at which the force is applied. This model assumes that the leader's trajectory ${\rm p_L}(t)$ is fixed as a function of time and that the force acts to bring the robot's position ${\rm p_R}(t)$ closer to ${\rm p_L}(t)$. Note that the training process does *not* depend on the specific model one uses to "translate" the intended trajectory to an interaction force; if a different model is used, the algorithm can still be applied. As was mentioned above, we assume that only noisy measurements $\{\tilde{F}_{\rm e}(t) \mid t \in [0, T]\}$ of the interaction force (11) are available; see Fig. 2 (middle) for an example of a typical forcing pattern. Although the force $F_{\rm e}$ applied at the biped is computed by (11), the low-level controllers use feedback of $\tilde{F}_{\rm e}$; that is, the controller applies torques $u = \Gamma_r(x, \tilde{F}_{\rm e})$. Thus, the noise will be considered as part of the environment, an instance of which is given by

$$E = \big\{\{x_0, \tilde{F}_{\rm e}(t), {\rm p_L}(t)\} \mid t \in [0, T]\big\} \quad (12)$$

which belongs in the space $\mathcal{E}$ of 3-tuples composed by the initial conditions and the *functions* describing the desired trajectory and the measurements of the interaction forces.

Next, we describe the mechanism that defines the distribution $\mathcal{D}$ by which random environment instances (12) are sampled from the space $\mathcal{E}$ of possible environments. In more detail, the initial conditions in E are selected to be those of the straight walking fixed point with randomly selected yaw angle $\tilde{q}_1 = q_1 + w_q$ with $w_q \sim \mathcal{N}(0, \sigma_q)$. The measurements $\{\tilde{F}_{\rm e}(t) \mid t \in [0, T]\}$ of the force that are available to the low-level controllers are assumed to be corrupted by white noise; i.e., $\tilde{F}_{\rm e}(t) = F_{\rm e}(t) + w_F$, where $w_F \sim \mathcal{N}(0, \sigma_F)$ and $F_{\rm e}(t)$ is given by (11). The values used for σ_q and σ_F are 0.1 rad and $0.1 I_{3\times 3}$ N, respectively. Finally, the desired trajectories $\{{\rm p_L}(t) \mid t \in [0, T]\}$ are generated by smoothening sequences of line segments of equal length with randomly selected slopes from a uniform distribution over the set $[-15°, 15°]$.

Our choice to represent the desired trajectories ${\rm p_L}(t)$ as above is motivated by collaborative object transportation tasks that involve pairs of co-workers in which one assumes the role of the leader [14], [16]. In such tasks, the leader is assumed to know where the object must be transported, and typically plans a smooth trajectory towards the goal location, occasionally changing directions. By modeling ${\rm p_L}(t)$ this way, the supervisor learns how to interpret noisy measurements of interaction forces in terms of changing directions in the intended trajectory. Finally, note that a wide variety of functions ${\rm p_L}(t)$ can be generated this way; see Fig. 2 (right).

B. Policy Parameterization

To find a PAC-Bayes policy, we parameterize the space of policies using a NN architecture with weights $w \in \mathbb{R}^d$. The policy is a NN with $d = 689$ parameters which consists of an input layer, two hidden layers and an output layer with n_i, 10, 20, and n_o neurons in each layer, respectively. The hidden layers are activated using an exponential linear unit (ELU) activation function while the output layer is activated with the Softmax activation function, which assigns the gait primitive scores as outputs.

The NN receives at its input a set of partial observations $y \in \mathcal{Y}$ that capture relevant gait and interaction features. In more detail, let $[t_{k-1}, t_k]$ be the duration of the k-th stride. The gait features ζ are chosen to include the values at t_k of the heading angle q_1 and the angle $\theta(q)$ used in (10) together

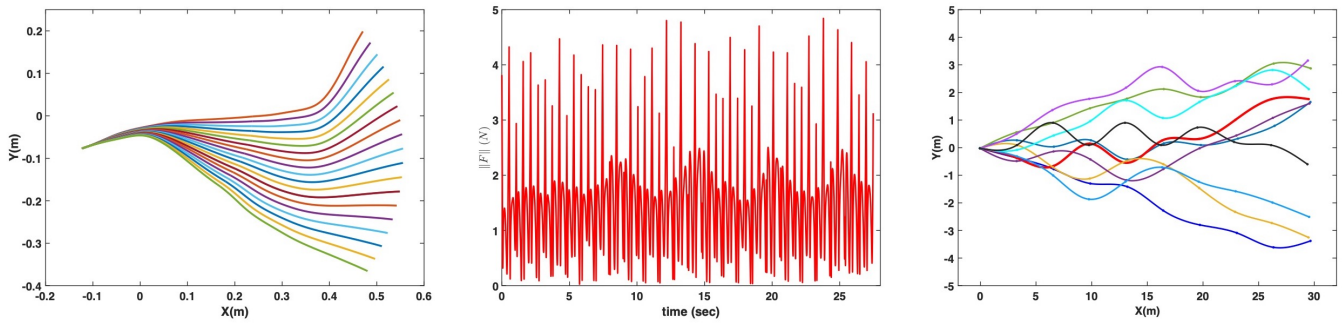

Fig. 2: (Left) Displacements (actions) in Cartesian space corresponding to the gaits in $\mathbb{G}$; no external force is considered here. (Middle) Magnitude of the noisy force $\tilde{F}_{\mathrm{e}}(t)$ for the bold red trajectory in the right figure; small magnitude indicates constructive interaction. (Right) Sampled trajectories $\mathrm{p_L}(t)$.

with their rates; i.e.[3], $\zeta = (q_1, \theta, \dot{q}_1, \dot{\theta})$. On the other hand, the interaction features are denoted by $(\Phi_{\mathrm{e}}^{\mathrm{x}}, \Phi_{\mathrm{e}}^{\mathrm{y}})$ and they correspond to integrals of the (noisy) measurements $(\tilde{F}_{\mathrm{e}}^{\mathrm{x}}, \tilde{F}_{\mathrm{e}}^{\mathrm{y}})$ of the x and y components of the interaction force over the duration $[t_{k-1}, t_k]$ of the k-th stride; i.e.,

$$\Phi_{\mathrm{e}}^{\mathrm{x}}(x; E) = \int_{t_{k-1}}^{t_k} \tilde{F}_{\mathrm{e}}^{\mathrm{x}}(t)dt \quad \text{and} \quad \Phi_{\mathrm{e}}^{\mathrm{y}}(x; E) = \int_{t_{k-1}}^{t_k} \tilde{F}_{\mathrm{e}}^{\mathrm{y}}(t)dt \ .$$

The NN processes these gait and interaction features and assigns a score to each gait primitive in $\mathbb{G}$, based on which the supervisor selects a suitable gait controller.

C. Training

The training pipeline consists of two stages. In the first stage, an inductive bias is extracted in the form of a probability distribution $\hat{P}$ over the space of policies. Loosely speaking, $\hat{P}$ reflects the "quality" of the policies generated by the NN. Choosing such distribution for NNs is often not intuitive; to remedy this problem, we adopt the approach in [27] and employ ES to compute $\hat{P}$ using a dataset $\hat{D} \sim \mathcal{D}^{\hat{N}}$ of $\hat{N}$ training environments. In the second stage, we leverage $\hat{P}$ to extract an informative prior distribution, which is then used to optimize the PAC-Bayes bound of Theorem 1. To do this, a dataset $D \sim \mathcal{D}^N$ of N training environments is used; the dataset D is sampled independently from $\hat{D}$.

1) Extracting inductive bias on the policy space: Our objective here is to uncover a useful inductive bias on the performance of the policies generated by the NN. Specifically, a probability distribution $\hat{P}$ will be obtained that is "peaked" around policies that, on average, perform well on environments drawn from $\mathcal{D}$. To evaluate the performance of a policy π applied in an environment E, we use the cost

$$\hat{C}(\pi; E) = \frac{1}{L} \int_0^T (e_{\mathrm{p}}^2(t) + e_{\phi}^2(t)) \ dt \tag{13}$$

in which L is the distance traveled by the leader over the interval $[0, T]$ and e_{p} and e_{ϕ} correspond to position and orientation errors [34, Chapter 5]. In more detail,

$$e_{\mathrm{p}} = \|\mathrm{p_L} - \mathrm{p_R}\|$$
$$e_{\phi} = \sqrt{(\cos\phi_{\mathrm{L}} - \cos\phi_{\mathrm{R}})^2 + (\sin\phi_{\mathrm{L}} - \sin\phi_{\mathrm{R}})^2}$$

[3]Intuitively, $(\theta, \dot{\theta})$ provide information about the robot's stride length and frequency, and leg $(q_1, \dot{q}_1)$ about its heading.

where dependence on t has been omitted, and ϕ_{L} and ϕ_{R} are the slopes of $\mathrm{p_L}$ and $\mathrm{p_R}$; see Fig. 1. Essentially, (13) assesses the "quality" of a policy based on the error of the biped's position and orientation from the intended trajectory.

To obtain the distribution $\hat{P}$, we will restrict our attention to the family of multivariate Gaussian distributions with diagonal covariance matrices; let $\mu \in \mathbb{R}^d$ be the mean and $\sigma \in \mathbb{R}^d$ the square root of the diagonal elements of the covariance matrix $\Sigma \in \mathbb{R}^{d \times d}$. Then, we sample $\hat{N}$ environments $\hat{D}$ from $\mathcal{E}$ and minimize the resulting empirical cost $\hat{C}_{\hat{D}}(\hat{P})$ with respect to the parameters $\psi := (\mu, \sigma) \in \mathbb{R}^{2d}$. The process requires the computation of the gradient

$$\nabla_{\psi} \hat{C}_{\hat{D}}(\hat{P}) = \frac{1}{\hat{N}} \sum_{E \in \hat{D}} \nabla_{\psi} \mathop{\mathbb{E}}_{w \sim \hat{P}} [\hat{C}(\pi_w; E)]$$

which, following [27], [35] can be decomposed as

$$\nabla_{\mu} \mathop{\mathbb{E}}_{w \sim \hat{P}} [\hat{C}(\pi_w; E)] = \mathop{\mathbb{E}}_{\epsilon \sim \mathcal{N}(0,I)} [\hat{C}(\mu + \sigma \odot \epsilon; E)\epsilon] \oslash \sigma$$
$$\nabla_{\sigma} \mathop{\mathbb{E}}_{w \sim \hat{P}} [\hat{C}(\pi_w; E)] = \mathop{\mathbb{E}}_{\epsilon \sim \mathcal{N}(0,I)} [\hat{C}(\mu + \sigma \odot \epsilon; E)(\epsilon \odot \epsilon - \mathbf{1})] \oslash \sigma$$

where $\odot$ and $\oslash$ denote the Hadamard (elementwise) product and division, respectively, and $\mathbf{1}$ is the d-dimensional vector with 1s. These expressions are used to estimate the gradient using Monte Carlo simulations over $2\hat{m}$ policies sampled from $\mathcal{N}(\mu, \Sigma)$. More details regarding the implementation of ES can be found in [27], [35]. We only mention here that we use antithetic sampling to reduce variance in the gradient estimates; that is, we always sample policies in ϵ and $-\epsilon$ pairs as detailed in [27]. With these estimates, the parameters ψ are updated in a gradient descent fashion according to $\psi_{t+1} \leftarrow \psi_t - \eta \nabla_{\psi} C_{\hat{D}}(\hat{P})$, where η is the learning rate. The outcome of the training process for the distribution $\hat{P}$ is the values $\hat{\psi} = (\hat{\mu}, \hat{\sigma})$, which will be used to extract an informative prior for optimizing the PAC-Bayes bounds (7).

2) Computing the PAC-Bayes policy: To provide an intuitive interpretation of the PAC-Bayes bound (7), we define a cost function that penalizes policies based on the fraction of the interval $[0, T]$ over which the biped violates a tube of radius $\mathrm{r} > 0$ around the intended trajectory $\mathrm{p_L}(t)$. In other words, the more a switching policy causes the biped to wander outside a pre-specified tube around the desired trajectory, the larger the cost associated with that policy is. Mathematically,

$$C(\pi, E) = \frac{1}{T} \int_0^T \mathbb{1}_{\{\|e_{\mathrm{p}}\| \geq \mathrm{r}\}} \ dt \tag{14}$$

where $\|e_{\mathrm{p}}\| = \|\mathrm{p_L} - \mathrm{p_R}\|$, and $\mathbb{1}_A$ denotes the indicator function for a given subset A. By definition, $C \in [0, 1]$.

Next, a suitable prior P_0 must be selected for the purpose of optimizing (7). One possible choice for P_0 is the distribution $\hat{P}$, which favors policies that perform better according to (13). Indeed, by Remark 1, using different cost functions for extracting an informative prior and for establishing the PAC-Bayes generalization guarantees is possible. However, choosing $\hat{P}$ as P_0 and minimizing the bound (7) with respect to the distribution P results in a high-dimensional optimization problem over the policy space $\mathbb{R}^d$; here $d = 689$.

To overcome this issue, we discretize the policy space $\mathbb{R}^d$ by sampling m policies from $\mathbb{R}^d$ according to $\hat{P}$ in an i.i.d. fashion. This way, inductive bias is embedded in the resulting *finite* policy space $\Pi := \{\pi_1, \pi_2, ..., \pi_m\}$, which will be used to optimize the PAC-Bayes bounds (7). In this setting, we choose as the prior distribution[4] p_0 over Π to be the (discrete) uniform distribution. Working with discrete distributions has the benefit that the KL divergence is a convex function, allowing us to express the resulting optimization as a convex program [27]. Furthermore, discrete distributions result in a significantly lower complexity for the space of policy distributions, thus yielding tighter bounds.

Now, to optimize the PAC-Bayes bound (7), we sample N environments $D = \{E_i\}_{i=1}^N$ from $\mathcal{E}$ according to $\mathcal{D}$. For each E_i, we evaluate the cost associated with each policy π_j in Π and form the cost matrix $C_{ij} = C(\pi_j; E_i)$, with $1 \le i \le N$ and $1 \le j \le m$. Thus, the average cost $C_j = \frac{1}{N}\sum_{i=1}^N C_{ij}$ of deploying policy π_j across all environments in D can be stacked to form a cost vector $\bar{C} \in \mathbb{R}^m$. Then, $C_D(p)$ in (7) can be expressed linearly in p as $\bar{C}p$, and following [27], minimizing the upper bound $C_{\mathrm{QPAC}}(p)$ can be written as

$$\min_{p\in\mathbb{R}^m} \left(\sqrt{\bar{C}p + R(p, p_0)} + \sqrt{R(p, p_0)}\right)^2 \tag{15}$$

$$\text{subject to} \quad \sum_{i=1}^m p_i = 1,\ 0 \le p_i \le 1 \tag{16}$$

where $R(p, p_0)$ is computed by (8) using p and p_0 and closed-form expressions for KL divergence. Following [26], (15)-(16) can be converted to a relative entropy program, an efficiently solvable class of convex programs.

D. Results and Interpretation

In the first stage of the proposed training pipeline, the distribution $\hat{P}$ is obtained using ES as explained in Section V-C1; the relevant hyperparameters are given in Table I. The ψ parameter updates are performed on minibatches of size 20 out of the $\hat{N} = 500$ environments that are used. Then, $\hat{P}$ is used to embed inductive bias in reducing the policy space to the finite collection Π of $m = 20$ policies. To obtain the PAC-Bayes bound, we introduce a tube of radius $\mathrm{r} = 0.5\mathrm{m}$ around $\mathrm{p_L}(t)$ for computing (14) and apply Theorem 1 in a discrete probability setting, as explained in Section V-C2. We select p_0 to be the uniform probability distribution over the reduced (finite) policy space Π and choose $\delta = 0.01$. The optimized PAC-Bayes bound for different numbers of training environments are given in Table II. Obtaining[5] a meaningful $\hat{P}$ is the most challenging task in terms of computational time. Training the PAC-Bayes policy takes ∼70 hours to compute the cost matrix $C_{ij} = C(\pi_j; E_i)$, with $1 \le i \le N$ and $1 \le j \le m$ on $N = 1000$ environments with $m = 20$ policy samples and ∼1 sec to solve (15)-(16). These calculations are performed offline. Equipped with a policy distribution $p^\star$ and the corresponding weights of the NN, the supervisor can suggest a suitable gait within a fraction of a millisecond by a forward pass through the NN.

TABLE I: Hyperparameters used in our two-stage training pipeline

Inductive bias					**PAC-Bayes**
Initial Dist.	#Envs $\hat{N}$	# ϵ $\hat{m}$	learning rate (η) μ	$\log(\sigma \odot \sigma)$	# Policy m
$\mathcal{N}(0, I)$	500	2	0.1	0.01	20

Table II also presents estimates of the true cost obtained by simulating the learned policy on 1000 novel environments; that is, environments that were not part of training. It can be seen that with increasing number of environments the PAC-Bayes bounds get closer to the empirical estimate of the true cost. To interpret the PAC-bounds presented, consider the last row of the Table II. According to Theorem 1, the biped tracks the leader's trajectory while staying in the tube 91.38% of the times with confidence 99%. Fig. 3 depicts two examples of applying the learned policy in novel environments.

TABLE II: PAC-Bayes costs for the interaction task

# Envs (N)	**PAC-bound** $(1 - C_{\mathrm{QPAC}}) \times 100$	**True success** (estimate)
200	82.97%	95.57%
500	89.19%	95.58%
1000	91.38%	95.47%

VI. Conclusion

We presented an approach to train a supervisor for gait adaptation for dynamically walking bipedal robots tasked with following a leader's unknown intended trajectory based on interaction. The supervisor is trained to orchestrate switching among a family of gait primitives by minimization of the PAC-Bayes upper bound. This way, the supervisor provides guarantees of generalization, essentially quantifying the risk of deploying a policy to novel leader intentions. We demonstrated the efficacy of our approach in deriving practical and strong generalization bounds in the case of a dynamic bipedal robot physically collaborating with a leader.

References

[1] K. Harada, E. Yoshida, and K. Yokoi, Eds., *Motion Planning for Humanoid Robots*. Springer-Verlag, 2010.

[2] P. Evrard and A. Kheddar, "Human-humanoid co-working in a joint table transportation," in *Proc. of the 4th Int. Conf. on Social Robotics*. Springer-Verlag, 2012, p. 357–366.

[4]Notation: Continuous probability distributions are denoted by uppercase letters while their discrete counterparts are denoted by lowercase letters.

[5]Training was performed on a desktop with 3.5 GHz Xeon W-2265 CPU, 12 cores, 64 GB RAM, and a 16 GB NVIDIA Quadro RTX 5000 GPU.

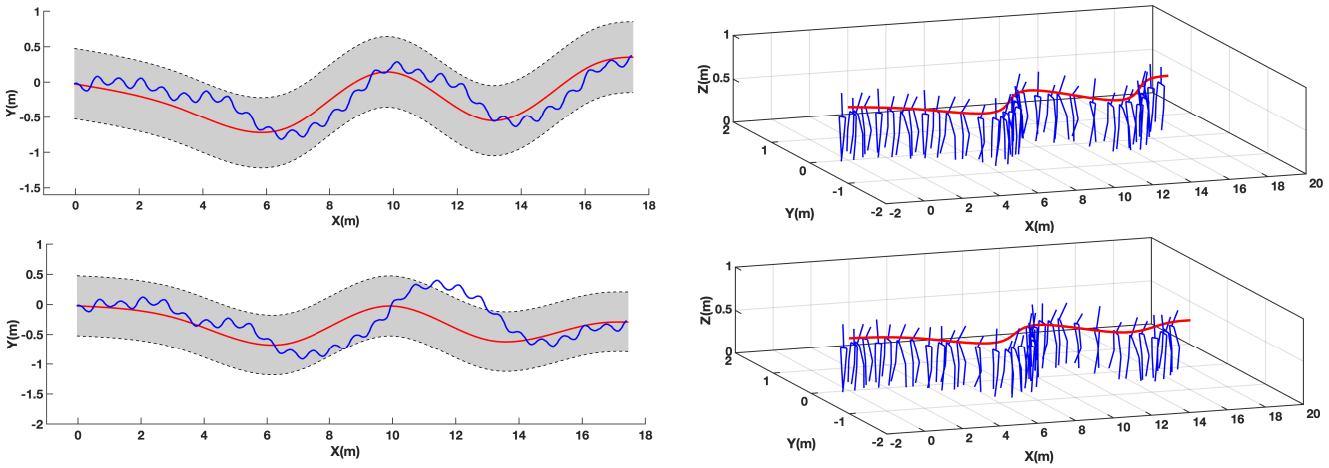

Fig. 3: Tracking performance of the biped under a learned policy. (Left) The intended trajectory $\mathrm{p_L}(t)$ is in red, the biped's trajectory $\mathrm{p_R}(t)$ is in blue and the tube of radius $r = 0.5\mathrm{m}$ around $\mathrm{p_L}(t)$ is in grey. (Right) Snapshots of the biped for the environments in the left. In the top environment, the biped successfully tracks the leader's trajectory while in the bottom environment the biped fails to stay within the tube.

[3] A. Bussy, A. Kheddar, A. Crosnier, and F. Keith, "Human-humanoid haptic joint object transportation case study," in *Proc. of the IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2012, pp. 3633–3638.

[4] E. Berger, D. Vogt, N. Haji-Ghassemi, B. Jung, and H. Ben Amor, "Inferring guidance information in cooperative human-robot tasks," in *Proc. of the IEEE Conf. on Humanoid Robots*, 2013, pp. 124–129.

[5] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback Control of Dynamic Bipedal Robot Locomotion*. Boca Raton, FL: CRC Press, 2007.

[6] J. Reher, C. Kann, and A. D. Ames, "An inverse dynamics approach to control Lyapunov functions," in *American Control Conf.*, 2020.

[7] G. A. Castillo, B. Weng, W. Zhang, and A. Hereid, "Robust feedback motion policy design using reinforcement learningon a 3D Digit bipedal robot," *arXiv:2013.15309v1*, 2021.

[8] M. S. Motahar, S. Veer, and I. Poulakakis, "Composing limit cycles for motion planning of 3D bipedal walkers," in *Proc. of IEEE Conf. on Decision and Control*, 2016, pp. 6368–6374.

[9] S. Veer, M. S. Motahar, and I. Poulakakis, "Almost driftless navigation of 3D limit-cycle walking bipeds," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 5025–5030.

[10] S. Teng, Y. Gong, J. W. Grizzle, and M. Ghaffari, "Toward safety-aware informative motion planning for legged robots," *arXiv:2103.14252v1*, 2021.

[11] X. Xiong, R. Reher, and A. D. Ames, "Global position control on underactuated bipedal robots: Step-to-step dynamics approximation for step planning," *arXiv:2011.06050v1*, 2020.

[12] J. Siekmann, K. Green, J. Warila, A. Fern, and J. Hurst, "Blind Bipedal Stair Traversal via Sim-to-Real Reinforcement Learning," in *Proceedings of Robotics: Science and Systems*, Virtual, July 2021.

[13] S. Veer, M. S. Motahar, and I. Poulakakis, "On the adaptation of dynamic walking to persistent external forcing using hybrid zero dynamics control," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2015, pp. 997–1003.

[14] M. S. Motahar, S. Veer, and I. Poulakakis, "Steering a 3D limit-cycle walker for collaboration with a leader," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 5251–5256.

[15] S. Veer, M. S. Motahar, and I. Poulakakis, "Adaptation of limit-cycle walkers for collaborative tasks: A supervisory switching control approach," in *Proc. of IEEE/RSJ Int. Conf. on Intelligent Robots and Systems*, 2017, pp. 5840–5845.

[16] S. Veer and I. Poulakakis, "Safe adaptive switching among dynamical movement primitives: Application to 3D limit-cycle walkers," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 2019.

[17] S. Veer and I. Poulakakis, "Switched systems with multiple equilibria under disturbances: Boundedness and practical stability," *IEEE Transactions on Automatic Control*, vol. 65, no. 6, pp. 2371–2386, 2020.

[18] A. Keemink, H. van der Kooij, and A. Stienen, "Admittance control for physical human-robot interaction," *Int. J. of Robotics Research*, vol. 37, no. 11, pp. 1421–1444, 2018.

[19] R. Ikeura and H. Inooka, "Variable impedance control of a robot for cooperation with a human," in *Proc. of the IEEE Int. Conf. on Robotics and Automation*, 1995, pp. 3097–3102.

[20] V. Duchaine and C. M. Gosselin, "General model of human-robot cooperation using a novel velocity based variable impedance control," in *EuroHaptics Conf. and Symp. on Haptic Interfaces for Virtual Environment and Teleoperator Systems*, 2007.

[21] J. Bae, K. Kim, J. Huh, and D. Hong, "Variable admittance control with virtual stiffness guidance for human-robot collaboration," *IEEE Access*, vol. 8, pp. 117 335–117 346, 2020.

[22] E. Gribovskaya, A. Kheddar, and A. Billard, "Motion learning and adaptive impedance for robot control during physical interaction with humans," in *2011 IEEE Int. Conf. on Robotics and Automation*, 2011, pp. 4326–4332.

[23] Y. Li and S. S. Ge, "Human-robot collaboration based on motion intention estimation," *IEEE/ASME Tr. on Mechatronics*, vol. 19, no. 3, pp. 1007–1014, 2014.

[24] I. Ranatunga, S. Cremer, D. O. Popa, and F. L. Lewis, "Intent aware adaptive admittance control for physical human-robot interaction," in *Proc. of the IEEE Int. Conf on Robotics and Automation*, 2015.

[25] O. Rivasplata, V. M. Tankasali, and C. Szepesvari, "PAC-Bayes with backprop," *arXiv:1908.07380*, 2019.

[26] A. Majumdar, A. Farid, and A. Sonar, "PAC-Bayes control: learning policies that provably generalize to novel environments," *Int. J. of Robotics Research*, vol. 40, no. 2-3, pp. 574–593, 2020.

[27] S. Veer and A. Majumdar, "Probably approximately correct vision-based planning using motion primitives," *arXiv:2002.12852*, 2020.

[28] A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, and S. Schaal, "Dynamical movement primitives: learning attractor models for motor behaviors," *Neural computation*, vol. 25, no. 2, pp. 328–373, 2013.

[29] S. Veer, Rakesh, and I. Poulakakis, "Input-to-state stability of periodic orbits of systems with impulse effects via Poincaré analysis," *IEEE Tr. on Automatic Control*, vol. 64, no. 11, pp. 4583–4598, 2019.

[30] D. A. McAllester, "Some PAC-Bayesian theorems," *Machine Learning*, vol. 37, no. 3, pp. 355–363, 1999.

[31] G. K. Dziugaite and D. M. Roy, "Computing nonvacuous generalization bounds for deep (stochastic) neural networks with many more parameters than training data," *arXiv:1703.11008*, 2017.

[32] C.-L. Shih, J. Grizzle, and C. Chevallereau, "From stable walking to steering of a 3d bipedal robot with passive point feet," *Robotica*, vol. 30, no. 07, pp. 1119–1130, 2012.

[33] C. Chevallereau, J. W. Grizzle, and C.-L. Shih, "Asymptotically stable walking of a five-link underactuated 3-d bipedal robot," *IEEE transactions on robotics*, vol. 25, no. 1, pp. 37–50, 2009.

[34] S. M. LaValle, *Planning Algorithms*. Cambridge, 2006.

[35] D. Wierstra, T. Schaul, T. Glasmachers, Y. Sun, J. Peters, and J. Schmidhuber, "Natural evolution strategies," *J. of Machine Learning Research*, vol. 15, no. 1, pp. 949–980, 2014.