Learning Controller Gains on Bipedal Walking Robots via User Preferences

Noel Csomay-Shanklin[1], Maegan Tucker[2], Min Dai[2], Jenna Reher[2], Aaron D. Ames[1,2]

Abstract— **Experimental demonstration of complex robotic behaviors relies heavily on finding the correct controller gains. This painstaking process is often completed by a domain expert, requiring deep knowledge of the relationship between parameter values and the resulting behavior of the system. Even when such knowledge is possessed, it can take significant effort to navigate the nonintuitive landscape of possible parameter combinations. In this work, we explore the extent to which preference-based learning can be used to optimize controller gains online by repeatedly querying the user for their preferences. This general methodology is applied to two variants of control Lyapunov function based nonlinear controllers framed as quadratic programs, which have nice theoretic properties but are challenging to realize in practice. These controllers are successfully demonstrated both on the planar underactuated biped, AMBER, and on the 3D underactuated biped, Cassie. We experimentally evaluate the performance of the learned controllers and show that the proposed method is repeatably able to learn gains that yield stable and robust locomotion.**

I. Introduction

Achieving robust and stable performance for physical robotic systems relies heavily on careful gain tuning, regardless of the implemented controller. Navigating the space of possible parameter combinations is a challenging endeavor, even for domain experts. To combat this challenge, researchers have developed systematic ways to tune gains for specific controller types [1]–[5]. For controllers where the input/output relationship between parameters and the resulting behavior is less clear, this can be prohibitively difficult. These difficulties are especially prevalent in the setting of bipedal locomotion, due to the extreme sensitivity of the stability of the system with respect to controller gains.

It was shown in [6] that control Lyapunov functions (CLFs) are capable of stabilizing locomotion through the hybrid zero dynamics (HZD) framework, with [7] demonstrating how this can be implemented as a quadratic program (QP), allowing the problem to be solved in a pointwise-optimal fashion even in the face of feasibility constraints. However, achieving robust walking behavior on physical bipeds can be difficult due to complexities such as compliance, under-actuation, and narrow domains of attraction. One such controller that has recently demonstrated stable locomotion on the 22 degree of freedom (DOF) Cassie biped, as shown in Figure 1, is the ID-CLF-QP$^+$ [8].

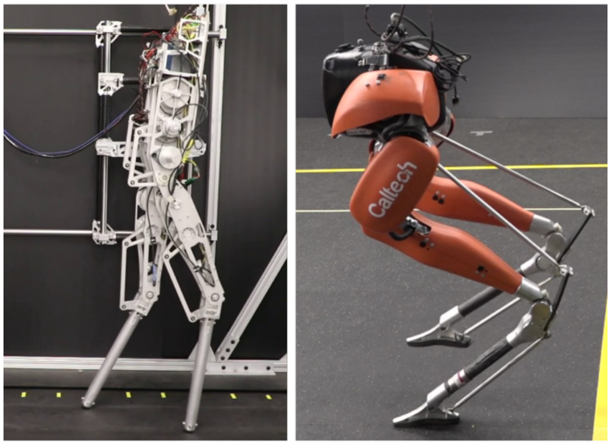


Fig. 1: The two experimental platforms investigated in this work: the planar AMBER-3M point-foot [9] robot (left), and the 3D Cassie robot [10] (right).

Synthesizing a controller capable of accounting for the complexities of underactuated locomotion, such as the ID-CLF-QP$^+$, necessitates the addition of numerous control parameters, exacerbating the issue of gain tuning. The relationship between the control parameters and the resulting behavior of the robot is extremely nonintuitive and results in a landscape that requires dedicated time to navigate, even for domain experts. For example, the implementation of the ID-CLF-QP$^+$ in [8] entailed 2 dedicated months of hand-tuning around 60 control parameters.

Recently, machine learning techniques have been implemented to alleviate the process of hand-tuning gains in a controller agnostic way by systematically navigating the entire parameter space [11]–[13]. However, these techniques rely on a carefully constructed predefined reward function. Moreover, it is often the case where different desired properties of the robotic behavior are conflicting such that they both can't be optimized simultaneously.

To alleviate the gain tuning process and enable the use of complicated controllers for naïve users, we propose a preference-based learning framework that only relies on subjective user feedback, mainly pairwise preferences, to systematically search the parameter space and realize stable and robust experimental walking. Preferences are a particularly useful feedback mechanism for parameter tuning because they are able to capture the notion of "general goodness" without a predefined reward function. Preference-based learning has been previously used towards selecting essential constraints of an HZD gait generation framework which resulted in stable and robust experimental walking on

This research was supported by NSF NRI award 1924526, NSF award 1932091, NSF CMMI award 1923239, NSF Graduate Research Fellowship No. DGE-1745301, and the Caltech Big Ideas and ZEITLIN Funds.

[1] Authors are with the Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125.

[2] Authors are with the Department of Mechanical and Civil Engineering, California Institute of Technology, Pasadena, CA 91125.

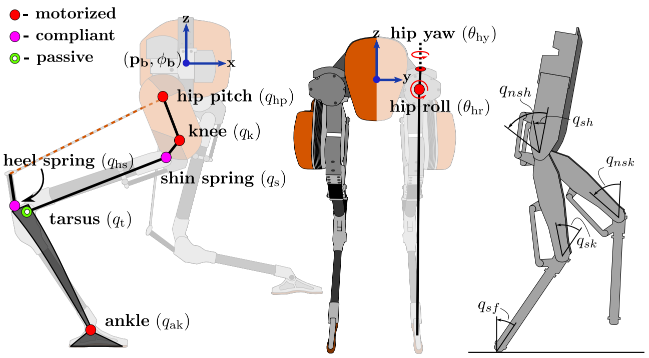

- motorized
- compliant
- passive
z
x
$(\mathbf{p_b}, \phi_\mathbf{b})$
hip pitch (q_{hp})
knee (q_{k})
heel spring (q_{hs})
shin spring (q_{s})
tarsus (q_{t})
ankle (q_{ak})
z
y
hip yaw (θ_{hy})
hip roll (θ_{hr})
q_{nsh}
q_{sh}
q_{nsk}
q_{sk}
q_{sf}

Fig. 2: Configuration of the 22 DOF (floating base) Cassie robot [10] (left) and configuration of the 5 DOF (pinned model) planar robot AMBER-3M [9] (right).

a planar biped with unmodeled compliance at the ankle [14].

In this work, we apply a similar preference-based learning framework as [14] towards learning gains of a CLF-QP$^+$ controller on the AMBER bipedal robot, as well as an ID-CLF-QP$^+$ controller on the Cassie bipedal robot. This application requires extending the learning framework to a much higher-dimensional space which led to unique challenges. First, more user feedback was required to navigate the larger action space. This was accomplished by sampling actions continuously on hardware which led to more efficient feedback collection. Second, to increase the speed of the learning, ordinal labels were also added as a feedback mechanism.

II. Preliminaries on Dynamics and Control

A. Modeling and Gait Generation

Following a floating-base convention [15], we begin with a general definition of a bipedal robot as a branched-chain collection of rigid linkages subjected to intermittent contact with the environment. We define the configuration space as $\mathcal{Q} \subset \mathbb{R}^n$, where n is the unconstrained DOF (degrees of freedom). Let $q = (p_b, \phi_b, q_l) \in \mathcal{Q} := \mathbb{R}^3 \times SO(3) \times \mathcal{Q}_l$, where p_b is the global Cartesian position of the body fixed frame attached to the base linkage (the pelvis), ϕ_b is its global orientation, and $q_l \in \mathcal{Q}_l \in \mathbb{R}^{n_l}$ are the local coordinates representing rotational joint angles. Further, the state space $\mathcal{X} = T\mathcal{Q} \subset \mathbb{R}^{2n}$ has coordinates $x = (q^\top, \dot{q}^\top)^\top$. The robot is subject to various *holonomic constraints*, which can be summarized by an equality constraint $h(q) \equiv 0$ where $h(q) \in \mathbb{R}^h$. Differentiating $h(q)$ twice and applying D'Alembert's principle to the Euler-Lagrange equations for the constrained system, the dynamics can be written as:

$$D(q)\ddot{q} + H(q, \dot{q}) = Bu + J(q)^\top \lambda \tag{1}$$

$$J(q)\ddot{q} + \dot{J}(q, \dot{q})\dot{q} = 0 \tag{2}$$

where $D(q) \in \mathbb{R}^{n \times n}$ is the mass-inertia matrix, $H(q, \dot{q})$ contains the Coriolis, gravity, and additional non-conservative forces, $B \in \mathbb{R}^{n \times m}$ is the actuation matrix, $J(q) \in \mathbb{R}^{h \times n}$ is the Jacobian matrix of the holonomic constraint, and $\lambda \in \mathbb{R}^h$ is the constraint wrench. The system of equations (1) for the dynamics can also be written in the control-affine form:

$$\dot{x} = \underbrace{\begin{bmatrix} \dot{q} \\ -D(q)^{-1}(H(q,\dot{q}) - J(q)^\top \lambda) \end{bmatrix}}_{f(x)} + \underbrace{\begin{bmatrix} 0 \\ D(q)^{-1}B \end{bmatrix}}_{g(x)} u.$$

The mappings $f : T\mathcal{Q} \to \mathbb{R}^n$ and $g : T\mathcal{Q} \to \mathbb{R}^{n \times m}$ are assumed to be locally Lipschitz continuous.

Dynamic and underactuated walking consists of periods of continuous motion followed by discrete impacts, which can be accurately modeled within a hybrid framework [16]. If we consider a bipedal robot undergoing domains of motion with only one foot in contact (either the left (L) or right (R)), and domain transition triggered at footstrike, then we can define:

$$\mathcal{D}_{SS}^{\{L,R\}} = \{(q, \dot{q}) : p_{swf}^z(q) \geq 0\},$$
$$\mathcal{S}_{L \to R, R \to L} = \{(q, \dot{q}) : p_{swf}^z(q) = 0, \dot{p}_{swf}^z(q, \dot{q}) < 0\},$$

where $p_{swf}^z : \mathcal{Q} \to \mathbb{R}$ is the vertical position of the swing foot, $\mathcal{D}_{SS}^{\{L,R\}}$ is the continuous domain on which our dynamics (1) evolve, with a transition from one stance leg to the next triggered by the switching surface $\mathcal{S}_{L \to R, R \to L}$. When this domain transition is triggered, the robot undergoes an impact with the ground, yielding a hybrid model:

$$\mathcal{HC} = \begin{cases} \dot{x} = f(x) + g(x)u & x \notin \mathcal{S}_{L \to R, R \to L} \\ \dot{x}^+ = \Delta(x^-) & x \in \mathcal{S}_{L \to R, R \to L} \end{cases} \tag{3}$$

where Δ is a plastic impact model [15] applied to the pre-impact states, x^-, such that the post-impact states, x^+, respect the holonomic constraints of the subsequent domain.

B. Hybrid Zero Dynamics

In this work, we design locomotion using the *hybrid zero dynamics* (HZD) framework [16], in order to design stable periodic walking for underactuated bipeds. At the core of this method is the regulation of *virtual constraints*, or outputs:

$$y(x) = y_a(x) - y_d(\tau, \alpha), \tag{4}$$

with the goal of driving $y \to 0$ where $y_a : T\mathcal{Q} \to \mathbb{R}^p$ and $y_d : T\mathcal{Q} \times \mathbb{R} \times \mathbb{R}^a \to \mathbb{R}^p$ are smooth functions, and α represents a set of Bezièr polynomial coefficients that can be shaped to encode stable locomotion.

If we assume the existence of a feedback controller $u^*(x)$ that can effectively stabilize this output tracking problem, then we can write the close-loop dynamics:

$$\dot{x} = f_{\mathrm{cl}}(x) = f(x) + g(x)u^*(x). \tag{5}$$

Additionally, by driving the outputs to zero this controller renders the *zero dynamics manifold*:

$$\mathcal{Z} = \{(q, \dot{q}) \in \mathcal{D} \mid y(x, \tau) = 0,\ L_{f_{\mathrm{cl}}} y(x, \tau) = 0\}. \tag{6}$$

forward invariant and attractive. However, because our system is represented as a hybrid system (3) we must also ensure that (6) is shaped such that the walking is stable through impact. We thus wish to enforce an impact invariance condition when we intersect with the switching surface:

$$\Delta(\mathcal{Z} \cap \mathcal{S}) \subset \mathcal{Z}. \tag{7}$$

In order to enforce this condition, the Bézier polynomials for the desired outputs can be shaped through the parameters α.

In order to generate walking behaviors using the HZD approach, we utilize the optimization library FROST [17] to transcribe the walking problem into an NLP:

$$\begin{aligned} (\alpha, \mathbf{X})^* = \underset{\alpha, \mathbf{X}}{\operatorname{argmin}} \quad & \mathcal{J}(\mathbf{X}) \\ \text{s.t.} \quad & \text{Closed loop dynamics (5)} \\ & \text{HZD condition (7)} \\ & \text{Physical feasibility} \end{aligned} \tag{8}$$

where $\mathbf{X} = (x_0, ..., x_N, T)$ is the collection of all decision variables with x_i the state at the i^{th} discretization and T the duration. The NLP (8) was solved with the optimizer IPOPT. This was done first for AMBER, in which one walking gait was designed using a pinned model of the robot [9], and then on Cassie for 3D locomotion using the motion library found in [18] consisting of 171 walking gaits for speeds in 0.1 m/s intervals on a grid for sagittal speeds of $v_x \in [-0.6, 1.2]$ m/s and coronal speeds of $v_y \in [-0.4, 0.4]$ m/s.

C. Control Lyapunov Functions

Control Lyapunov functions (CLFs), and specifically rapidly exponentially stabilizing control Lyapunov functions (RES-CLFs), were introduced as methods for achieving (rapidly) exponential stability on walking robots [19]. This control approach has the benefit of yielding a control framework that can provably stabilize periodic orbits for hybrid system models of walking robots, and can be realized in a pointwise optimal fashion. In this work, we consider only outputs which are *vector relative degree* 2. Thus, differentiating (4) twice with respect to the dynamics results in:

$$\ddot{y}(x) = L_f^2 y(x) + L_g L_f y(x) u.$$

Assuming that the system is feedback linearizeable, we can invert the decoupling matrix, $L_g L_f y(x)$, to construct a preliminary control input:

$$u = (L_g L_f y(x))^{-1} \left(\nu - L_f^2 y(x)\right), \tag{9}$$

which renders the output dynamics to be $\ddot{y} = \nu$. With the auxiliary input ν appropriately chosen, the nonlinear system can be made exponentially stable. Assuming the preliminary controller (9) has been applied to our system, and defining $\eta = [y_2, \dot{y}_2]^\top$ we have the following output dynamics [20]:

$$\dot{\eta} = \underbrace{\begin{bmatrix} 0 & I \\ 0 & 0 \end{bmatrix}}_{F} \eta + \underbrace{\begin{bmatrix} 0 \\ I \end{bmatrix}}_{G} v. \tag{10}$$

With the goal of constructing a CLF using (10), we evaluate the continuous time algebraic Ricatti equation (CARE):

$$F^\top P + PF + PGR^{-1}G^\top P + Q = 0, \tag{CARE}$$

which has a solution $P \succ 0$ for any $Q = Q^\top \succ 0$ and $R = R^\top \succ 0$. From the solution of (CARE), we can construct a rapidly exponentially stabilizing CLF (RES-CLF) [19]:

$$V(\eta) = \eta^\top \underbrace{I_\varepsilon P I_\varepsilon}_{P_\varepsilon} \eta, \qquad I_\varepsilon = \begin{bmatrix} \frac{1}{\varepsilon} I & 0 \\ 0 & I \end{bmatrix}, \tag{11}$$

which, for $0 < \varepsilon < 1$, is a tunable parameter that drives the (rapidly) exponential convergence. Any feedback controller, u, which can satisfy the convergence condition:

$$\begin{aligned} \dot{V}(\eta) &= L_F V(\eta) + L_G V(\eta) \nu \\ &= L_F V(\eta) + L_G V(\eta) \left(L_g L_f y(x) u + L_f^2 y(x)\right) \\ &= L_f V(\eta) + L_g V(\eta) u \leq -\frac{1}{\varepsilon} \underbrace{\frac{\lambda_{min}(Q)}{\lambda_{max}(P)}}_{\gamma} V(\eta), \end{aligned} \tag{12}$$

will then render rapidly exponential stability for the output dynamics (4). In the context of RES-CLF, we can then define:

$$K_\varepsilon(x) = \{u_\varepsilon \in U : L_f V(x) + L_g V(x) u + \frac{\gamma}{\varepsilon} V(x) \leq 0\},$$

describing an entire class of the controllers which result in (rapidly) exponential convergence. This leads naturally to the consideration of an optimization-based approach to enforcing (12). One such approach is to pose the CLF problem within a quadratic program (CLF-QP), with (12) as an inequality constraint [7]. When implementing this controller on physical systems, which are often subject to additional constraints such as torque limits or friction limits, a weighted relaxation term, δ, is added (12) in order to maintain feasibility.

CLF-QP-δ:

$$\begin{aligned} u^* = \underset{u \in \mathbb{R}^m}{\operatorname{argmin}} \quad & \|L_f^2 y(x) + L_g L_f y(x) u\|^2 + w_{\dot{V}} \delta^2 \\ \text{s.t.} \quad & \dot{V}(x) = L_f V(x) + L_g V(x) u \leq -\frac{\gamma}{\varepsilon} V + \delta \\ & u_{min} \preceq u \preceq u_{max} \end{aligned} \tag{13}$$

Because this relaxation term is penalized in the cost, we could also move the inequality constraint completely into the cost as an exact penalty function [8]:

$$\mathcal{J}_\delta = \|L_f^2 y(x) + L_g L_f y(x) u\|^2 + w_{\dot{V}} ||g^+(x, u)||$$

where:

$$\begin{aligned} g(x, u) &:= L_f V(x) + L_g V(x) u + \frac{\gamma}{\varepsilon} V(x), \\ g^+(x, u) &\triangleq \max(g, 0), \end{aligned}$$

One of the downsides to using this approach is that the cost term $||g^+(x, u)||$ will intermittently trigger and cause a jump to occur in the commanded torque. Instead, we can allow $g(x, u)$ to go negative, meaning that the controller will always drive convergence even when the inequality (12) is not triggered [21]. This leads to the following relaxed (CLF-QP) with incentivized convergence in the cost:

CLF-QP$^+$:

$$\begin{aligned} u^* = \underset{u \in \mathbb{R}^m}{\operatorname{argmin}} \quad & \|L_f^2 y(x) + L_g L_f y(x) u\|^2 + w_{\dot{V}} \dot{V}(x, u) \\ \text{s.t.} \quad & u_{min} \preceq u \preceq u_{max} \end{aligned} \tag{14}$$

In order to avoid computationally expensive inversions of the model sensitive mass-inertia matrix, and to allow for a variety of costs and constraints to be implemented, a variant of the (CLF-QP) termed the (ID-CLF-QP) was introduced in [21]. This controller is used on the Cassie biped, with the decision variables $\mathcal{X} = [\ddot{q}^\top, u^\top, \lambda^\top]^\top \in \mathbb{R}^{39}$:

ID-CLF-QP$^+$:

$$\mathcal{X}^* = \underset{\mathcal{X} \in \mathbb{X}_{ext}}{\operatorname{argmin}} \quad \|A(x)\mathcal{X} - b(x)\|^2 + \dot{V}(q, \dot{q}, \ddot{q}) \tag{15}$$

$$\begin{aligned} \text{s.t.} \quad & D(q)\ddot{q} + H(q, \dot{q}) = Bu + J(q)^\top \lambda \\ & u_{min} \preceq u \preceq u_{max} \\ & \lambda \in \mathcal{AC}(\mathcal{X}) \end{aligned} \tag{16}$$

where (2) has been moved into the cost terms $A(x)$ and $b(x)$ as a weighted soft constraint, in addition to a feedback linearizing cost, and a regularization for the nominal $\mathcal{X}^*(\tau)$ from the HZD optimization. Interested readers are referred to [8], [21] for the full (ID-CLF-QP+) formulation.

D. Parameterization of CLF-QP

For the following discussion, let $a = [a_1, ..., a_v] \in A \subset \mathbb{R}^v$ be an element of a $v-$dimensional parameter space, termed an *action*. We let $Q = Q(a)$, $\varepsilon = \varepsilon(a)$, and $w_{\dot{V}} = w_{\dot{V}}(a)$ denote a parameterization of our control tuning variables, which will subsequently be learned. Each gain a_i for $i = 1, \ldots, v$ is discretized into d_i values, leading to an overall search space of actions given by the set $\mathbf{A}$ with cardinality $|\mathbf{A}| = \prod_{i=1}^{v} d_i$. In this work, experiments are conducted on two separate experimental platforms: the planar biped AMBER, and the 3D biped Cassie. For AMBER, v is taken to be 6 with discretizations $d = [4, 4, 5, 5, 4, 5]$, resulting in the following parameterization:

$$\begin{aligned} Q(a) &= \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}, & Q_1 &= \operatorname{diag}([a_1, a_2, a_2, a_1]), \\ & & Q_2 &= \operatorname{diag}([a_3, a_4, a_4, a_3]), \\ \varepsilon(a) &= a_5, & w_{\dot{V}}(a) &= a_6, \end{aligned}$$

which satisfies $Q(a) \succ 0$, $0 < \varepsilon(a) < 1$, and $w_{\dot{V}}(a) > 0$ for the choice of bounds, as summarized in Table I. Because of the simplicity of AMBER, we were able to tune all associated gains for the CLF-QP$^+$ controller. For Cassie, however, the complexity of the ID-CLF-QP$^+$ controller warranted only a subset of parameters to be selected. Namely, v is taken to be 12 and d_i to be 8, resulting in:

$$Q = \begin{bmatrix} Q_1 & 0 \\ 0 & Q_2 \end{bmatrix}, \quad \begin{aligned} Q_1 &= \operatorname{diag}([a_1, \ldots, a_{12}]), \\ Q_2 &= \bar{Q}, \end{aligned}$$

with $\bar{Q}$, ε, and $w_{\dot{V}}$ remaining fixed and predetermined by a domain expert. From this definition of Q, we can split our output coordinates $\eta = (\eta_t, \eta_{nt})$ into *tuned* and *not-tuned* components, where $\eta_t \in \mathbb{R}^{12}$ and $\eta_{nt} \in \mathbb{R}^6$ correspond to the Q_1 and Q_2 blocks in in Q.

TABLE I: Learned Parameters

CASSIE		
	Pos. Bounds	Vel. Bounds
Q Pelvis Roll (ϕ_x)	a_1:[2000, 12000]	a_7:[5, 200]
Q Pelvis Pitch (ϕ_y)	a_2:[2000, 12000]	a_8:[5, 200]
Q Stance Leg Length ($\|\phi^{st}\|_2$)	a_3:[4000, 15000]	a_9:[50, 500]
Q Swing Leg Length ($\|\phi^{sw}\|_2$)	a_4:[4000, 20000]	a_{10}:[50, 500]
Q Swing Leg Angle (θ^{sw}_{hp})	a_5:[1000, 10000]	a_{11}:[10, 200]
Q Swing Leg Roll (θ^{sw}_{hr})	a_6:[1000, 8000]	a_{12}:[5, 150]

AMBER				
	Pos. Bounds	Vel. Bounds		Bounds
Q Knees	a_1:[100, 1500]	a_3:[10, 300]	ε	a_5:[0.08, 0.2]
Q Hips	a_2:[100, 1500]	a_4:[10, 300]	$w_{\dot{V}}$	a_6:[1, 5]

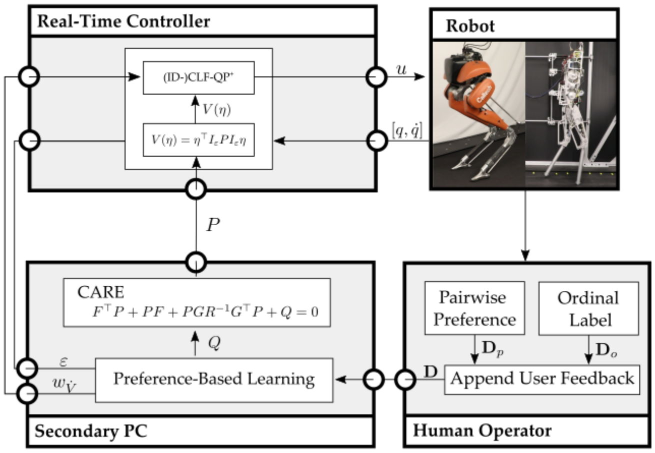


Fig. 3: The experimental procedure, notably the communication between the controller, physical robot, human operator, and learning framework.

III. Learning Framework

In this section we will present this preference-based learning framework used in this work, specifically aimed at tuning controller gains. We assume that the user has some unknown underlying utility function $U : \mathbf{A} \to \mathbb{R}$, which maps actions to a personal rating of how good of the experimental walking seems to them. The goal of the framework is to identify the user preferred action, $a^* = \operatorname{argmax}_a U(a)$, in as few iterations as possible.

In general, Bayesian optimization is a probabilistic approach towards identifying a^* by selecting $\hat{a}^*$, the action believed to be optimal, which minimizes $||\hat{a}^* - a^*||_2$. Typically, Bayesian optimization is used on problems where the underlying function is difficult to evaluate but can be obtained. Recent work extended Bayesian optimization to the preference setting [22], where the action maximizing the users underlying utility function $U(a)$ is obtained using only pairwise preferences between sampled actions. We refer to this setting as "preference-based learning". In this work, we utilize a more recent preference-based learning algorithm, LineCoSpar [23] with the addition of ordinal labels inspired from [24], which maintains the posterior only over a subset of the entire actions space to increase computation tractability – more details can be found in [14]. The resulting learning framework iteratively applies Thompson sampling to navigate a high-dimensional Bayesian landscape of user preferences.

A. Summary of Learning Method

A summary of the learning method is as follows. At each iteration, the user is queried for their preference between the most recently sampled action, a_i, and the previous action, a_{i-1}. We define a likelihood function based on preferences:

$$\mathcal{P}(a_i \succ a_{i-1}|U(a_i), U(a_{i-1})) = \begin{cases} 1 & \text{if } U(a_i) \geq U(a_{i-1}) \\ 0 & \text{otherwise,} \end{cases}$$

where $a_i \succ a_{i-1}$ denotes a preference of action a_i over action a_{i-1}. In other words, the likelihood function states that the user has utility $U(a_i) \geq U(a_{i-1})$ with probability 1 given that they return a preference $a_i \succ a_{i-1}$. This is a strong assumption on the ability of the user to give noise-free feedback; to account for noisy preferences we instead use:

$$\mathcal{P}(a_i \succ a_{i-1}|U(a_i), U(a_{i-1})) = \phi\left(\frac{U(a_i) - U(a_{i-1})}{c_p}\right),$$

where $\phi : \mathbb{R} \to (0, 1)$ is a monotonically-increasing link function, and $c_p > 0$ represents the amount of noise expected in the preferences. In this work, we select the heavy-tailed sigmoid distribution $\phi(x) := \frac{1}{1+e^{-x}}$.

Inspired by [24], we supplement preference feedback with ordinal labels. Because ordinal labels are expected to be noisy, the ordinal categories are limited to only "very bad", "neutral", and "very good". Ordinal labels are obtained each iteration for the corresponding action a_i and are assumed to be assigned based on $U(a_i)$. Just as with preferences, a likelihood function is created for ordinal labels:

$$\mathcal{P}(o = r|U(a_i)) = \begin{cases} 1 & \text{if } b_{r-1} < U(a_i) < b_r \\ 0 & \text{otherwise} \end{cases}$$

where $\{b_0, \dots, b_N\}$ are arbitrary thresholds that dictate which latent utility ranges correspond to which ordinal label assuming ideal noise-free feedback. In our work, these thresholds were selected to be $\{-\inf, -1, 1, \inf\}$. Again, the likelihood function is modified to account for noise by a link function ϕ and expected noise in the ordinal labels $c_o > 0$:

$$\mathcal{P}(o = r|U(a)) = \phi\left(\frac{b_r - U(a_m)}{c_o}\right) - \phi\left(\frac{b_{r-1} - U(a)}{c_o}\right).$$

After every sampled action a_i, the human operator is queried for both a pairwise preference between a_{i-1} and a_i as well as an ordinal label for a_i. This user feedback is added to respective datasets $\mathbf{D}_p = \{a_{k_1(i)} \succ a_{k_2(i)}, i = 1, \dots, n\}$, and $\mathbf{D}_o = \{o_i, i = 1, \dots, n\}$, with the total dataset of user feedback denoted as $\mathbf{D} = \mathbf{D}_p \cup \mathbf{D}_o$.

To infer the latent utilities of the sampled actions $\boldsymbol{U} = [U(a_1), \dots, U(a_N)]^\top$ using $\mathbf{D}$, we apply the preference-based Gaussian process model to the posterior distribution $\mathcal{P}(\boldsymbol{U}|\mathbf{D})$ as in [25]. First, we model the posterior distribution as proportional to the likelihoods multiplied by the Gaussian prior using Bayes rule,

$$\mathcal{P}(\boldsymbol{U}|\mathbf{D}_p, \mathbf{D}_o) \propto \mathcal{P}(\mathbf{D}_o, \mathbf{D}_p|\boldsymbol{U})\mathcal{P}(\boldsymbol{U}), \tag{17}$$

where the Gaussian prior over $\boldsymbol{U}$ is given by:

$$\mathcal{P}(\boldsymbol{U}) = \frac{1}{(2\pi)^{|\mathbf{V}|/2}|\Sigma|^{1/2}} \exp\left(-\frac{1}{2}\boldsymbol{U}^\top \Sigma^{-1} \boldsymbol{U}\right).$$

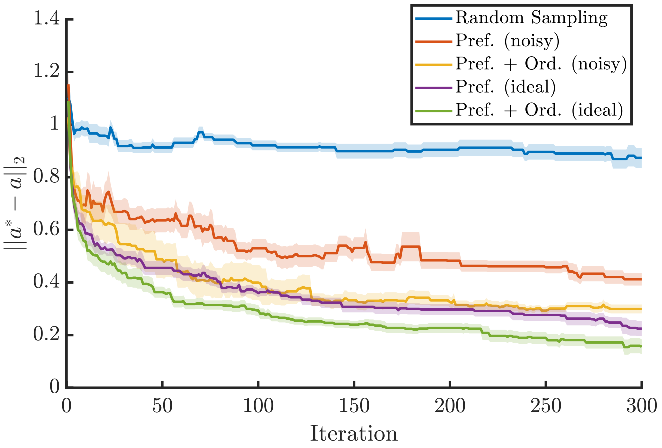


Fig. 4: Simulated learning results averaged over 10 runs, demonstrating the capability of preference-based learning to optimize over large action spaces, specifically the one used for experiments with Cassie. Standard error is shown by the shaded region.

with $\Sigma \in \mathbb{R}^{|\mathbf{V}| \times |\mathbf{V}|}$, $\Sigma_{ij} = \mathcal{K}(a_i, a_j)$, and $\mathcal{K}$ is a kernel. Assuming conditional independence of queries, we can split $\mathcal{P}(\mathbf{D}_o, \mathbf{D}_p|\boldsymbol{U}) = \mathcal{P}(\mathbf{D}_o|\boldsymbol{U})\mathcal{P}(\mathbf{D}_p|\boldsymbol{U})$ wherse

$$\mathcal{P}(\mathbf{D}_p|\boldsymbol{U}) = \prod_{i=1}^{K} \mathcal{P}(a_1 \succ a_2|U(a_1), U(a_2)),$$

$$\mathcal{P}(\mathbf{D}_o|\boldsymbol{U}) = \prod_{i=1}^{M} \mathcal{P}(a_1 = r_1|U(a_1)).$$

The posterior (17) is then estimated via the Laplace approximation as in [25], which yields a multivaraite Gaussian $\mathcal{N}(\mu, \sigma)$. The mean μ can be interpreted as our estimate of the latent utilities $\boldsymbol{U}$ with uncertainty $\sqrt{\text{diag}(\sigma^{-1})}$.

To select new actions to query in each iteration, we apply a Thomposon sampling approach. Specifically, at each iteration we draw a random sample from $U \sim \mathcal{N}(\mu, \Sigma)$ and select the action which maximizes U as:

$$a = \underset{a}{\text{argmax}}\, U(a). \tag{18}$$

This action is then given an ordinal label, and a preference is collected between it and the previous action. This process is completed for as many iterations as is desired. The best action after the iterations have been completed is

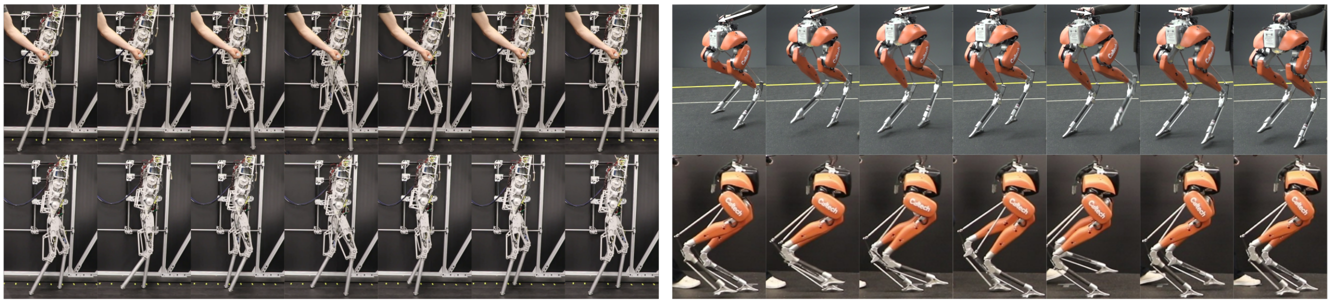
$$\hat{a}^* = \underset{a}{\text{argmax}}\, \mu(a)$$

where μ is the mean function of the multivariate Gaussian.

B. Expected Learning Behavior

To demonstrate the expected behavior of the learning algorithm, a toy example was constructed of the same dimensionality as the controller parameter space being investigated on Cassie ($v = 12, d = 8$), where the utility was modeled as $U(a) = \|a - a^*\|_2$ for some a^*. Feedback was automatically generated for both ideal noise-free feedback as well as for noisy feedback (correct feedback given with probability 0.9).

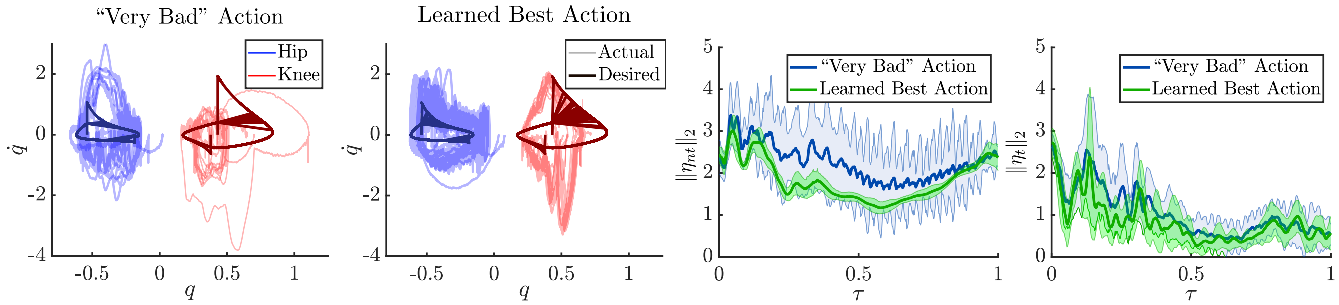
The results of the simulated algorithm, illustrated in Fig. 4, show that the learning framework is capable of decreasing



(a) The behavior corresponding to a very low utility (top) and to the maximum posterior utility (bottom).

(b) The robustness (top) and and tracking (bottom) of the walking with the learned optimal gains is demonstrated through gait tiles.

Fig. 5: Gait tiles for AMBER (left) and Cassie (right).



(a) Phase portraits for AMBER experiments.

(b) Output Error of η_t (left) and η_{nt} (right) for Cassie experiment.

Fig. 6: Experimental walking behavior of the CLF-QP$^+$ (left) and the ID-CLF-QP$^+$ (right) with the learned gains.

the error in the believed optimal action $\hat{a}^*$ even for an action space as large as the one used in the experiments with Cassie. The simulated results also show that ordinal labels allow for faster convergence to the optimal action, even in the case of noise, motivating their use in the final experiment. Lastly, the preference-based learning framework was also compared to random sampling, where the only difference in the algorithm was that actions were selected randomly. In comparison, the random sampling method leads to minimal improvement when compared to preference-based learning. From these simulation results, it can clearly be seen that the proposed method is an effective mechanism for exploring high-dimensional parameter spaces.

IV. Learning to Walk in Experiments

Preference-based learning applied to tuning control parameters was experimentally implemented on two separate robotic platforms: the 5 DOF planar biped AMBER, and the 22 DOF 3D biped Cassie, as can be seen in the video [26]. A visualization of the experimental procedure is given in Figure 3. The experiments had four main components: the physical robot (either AMBER or Cassie), the controller running on a real-time PC, a human operating the robot who gave their preferences, and a secondary PC running the learning algorithm. The user feedback provided to the learning algorithm included pairwise preferences and ordinal labels. For the pairwise preferences, the human operator was asked “Do you prefer this behavior more or less than the last behavior”. For the ordinal labels, the human was asked to provide a label of either “very bad, neutral, or very good”.

User feedback was obtained after each sampled action was experimentally deployed on the robot. Each action was tested for approximately 30 seconds to 1 minute, during which the behavior of the robot was evaluated in terms of both performance and robustness. After user feedback was collected for the sampled controller gains, the posterior was inferred over all of the uniquely sampled actions, which took up to 0.5 seconds. The experiment with AMBER was conducted for 50 iterations, lasing approximately one hour, and the experiment with Cassie was conducted for 100 iterations, lasting one hour for the domain expert and roughly two hours for the naïve user.

A. Results with AMBER

The preference-based learning framework is first demonstrated on tuning the gains associated with the CLF-QP$^+$ for the AMBER bipedal robot. The CLF-QP$^+$ controller was implemented on an off-board i7-6700HQ CPU @ 2.6GHz with 16 GB RAM, which solved for desired torques and communicated them with the ELMO motor drivers on the AMBER robot. The motor driver communication and CLF-QP$^+$ controller ran at 2kHz. During the first half of the experiment, the algorithm sampled a variety of gains causing behavior ranging from instantaneous torque chatter to induced tripping due to inferior output tracking. By the end of the experiment, the algorithm had sampled 3 gains which were deemed ”very good”, and which resulted in stable walking behavior. Gait tiles for an action deemed “very bad”, as well as the learned best action are shown in Figure 5a. Additionally, tracking performance for the two sets of gains

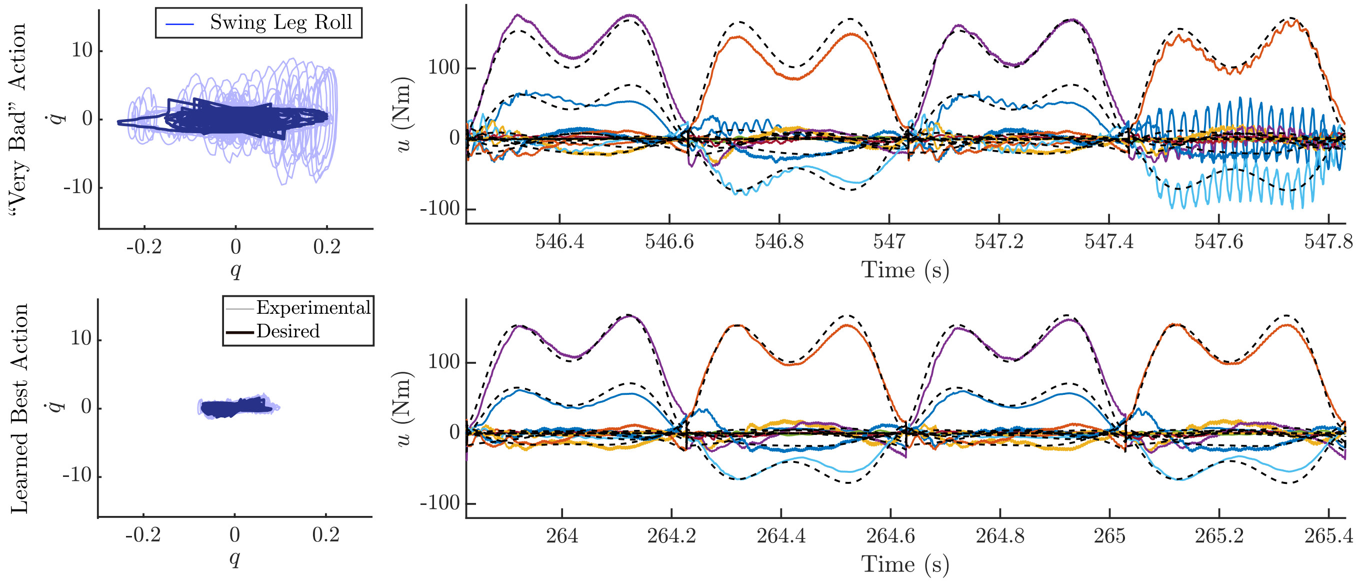


Fig. 7: Phase plots and torques commanded by the ID-CLF-QP$^+$ in the naïve user experiments with Cassie. For torques, each colored line corresponds to a different joint, with the black dotted lines being the feedforward torque. The gains corresponding to a "very bad" action (top) yield torques that exhibit poor tracking on joints and torque chatter. On the other hand, the gains corresponding to the learned optimal action (bottom) exhibit much better tracking and no torque chatter.

is seen in Figure 6a, where the learned best action tracks the desired behavior to a better degree.

The importance of the relative weight of the parameters can be seen by looking at the learned best action:

$$\hat{a}^* = [750, 100, 300, 100, 0.125, 2].$$

Interestingly, the knees are weighted higher than the hips in the Q matrix, which is reflected in the desired convergence of these outputs when constructing the the Lyapunov function. Also, the values of ε and $w_{\dot{V}}$ are in the middle of the given range, suggesting that undesirable behavior results from these values being too high or too low. In the end, applying preference-based learning to tuning the gains of the CLF-QP$^+$ on AMBER resulted in stable walking and in one of the few instantiations of a CLF-QP running on hardware.

B. Results with Cassie

To test the capability of the learning method towards tuning more complex controllers, the preference-based learning method was applied for tuning the gains of the ID-CLF-QP$^+$ controller for the Cassie bipedal robot. To demonstrate repeatability, the experiment was conducted twice: once with a domain expert, and once with a naïve user. In both experiments, a subset of the Q matrix from (CARE) was tuned with coarse bounds given by a domain export, as reported in Table I. These specific outputs were chosen because they were deemed to have a large impact on the performance of the controller. Additionally, the regularization terms in (15) were lowered when compared to the baseline controller for both experiments so that the effect of the outputs would be more noticeable. Although lower regularization terms encourage faster convergence of the outputs to the zero dynamics surface, they induce increased torque chatter and lead to a more challenging gain tuning process.

The controller was implemented on the on-board Intel NUC computer, which was running a PREEMPT_RT kernel. The software runs on two ROS nodes, one of which communicate state information and joint torques over UDP to the Simulink Real-Time xPC, and one of which runs the controller. Each node is given a separate core on the CPU, and is elevated to real-time priority. Preference-based learning was run on an external computer and was connected to the ROS master over wifi. Actions were updated continuously with no break in between each walking motion. To accomplish this real-time update, once an action was selected it was sent to Cassie via a rosservice call, where, upon receipt, the robot immediately updated the corresponding gains. Because rosservice calls are blocking, multithreading their receipt and parsing was necessary in order to maintain real-time performance.

For both experiments, preferences were dictated by the following criteria (ordered by importance): no torque chatter, no drift in the floating base frame, responsiveness to desired directional input, and no violent impacts. At the start of the experiments, there was significant torque chatter and wandering, with the user having to regularly intervene to recenter the global frame. As the experiments continued, the walking generally improved, but not strictly. At the conclusion of 100 iterations, the posterior was inferred over all uniquely visited actions. The action corresponding with the maximum utility – believed by the algorithm to result in the most user preferred walking behavior – was further evaluated for tracking and robustness. In the end, this learned best action coincided with the walking behavior that the user preferred the most, and the domain expert found the learned gains to be "objectively good". The optimal gains identified by the framework are:

$$\begin{aligned}\hat{a}^* =&[2400, 1700, 4200, 5600,\\ &1700, 1200, 27, 40, 120, 56, 17, 7].\end{aligned}$$

Features of this optimal action, compared to a worse action sampled in the beginning of the experiments, are outlined in Figure 6. In terms of quantifiable improvement, the difference in tracking performance is shown in Figure 6b. For the sake of presentation, the outputs are split into $\eta = (\eta_t,\ \eta_{nt})$ where η_t are the 12 outputs whose parameters were tuned by the learning algorithm and η_{nt} are the remaining 6 outputs. The magnitude of η_t illustrates the improvement that preference-based learning attained in tracking the outputs it intended to. At the same time, the tracking error of η_{nt} shows that the outputs that were not tuned remained unaffected by the learning process. This quantifiable improvement is further illustrated by the commanded torques in Figure 7, which show that the optimal gains result in much less torque chatter and better tracking as compared to the other gains.

Limitations. The main limitation of the current formulation of preference-based learning towards tuning controller gains is that the action space bounds must be predefined, and these bounds are often difficult to know *a priori*. Future work to address this problem involves modifications to the learning framework to shift action space based on the user's preferences. Furthermore, the current framework limits the set of potential new actions to the set of actions discretized by d_i for each dimension i. As such, future work also includes adapting the granularity of the action space based on the uncertainty in specific regions.

V. Conclusion

Navigating the complex landscape of controller gains is a challenging process that often requires significant knowledge and expertise. In this work, we demonstrated that preference-based learning is an effective mechanism towards systematically exploring a high-dimensional controller parameter space. Furthermore, we experimentally demonstrated the power of this method on two different platforms with two different controllers, showing the application agnostic nature of framework. In all experiments, the robots went from stumbling to walking in a matter of hours. Additionally, the learned best gains in both experiments corresponded with the walking trials most preferred by the human operator. In the end, the robots had improved tracking performance, and were robust to external disturbance. Future work includes addressing the aforementioned limitations, extending this methodology to other robotic platforms, coupling preference-based learning with metric-based optimization techniques, and addressing multi-layered parameter tuning tasks.

References

[1] L. Zheng, "A practical guide to tune of proportional and integral (pi) like fuzzy controllers," in *[1992 Proceedings] IEEE International Conference on Fuzzy Systems*. IEEE, 1992, pp. 633–640.

[2] Y. Zhao, W. Xie, and X. Tu, "Performance-based parameter tuning method of model-driven pid control systems," *ISA transactions*, vol. 51, no. 3, pp. 393–399, 2012.

[3] H. Hjalmarsson and T. Birkeland, "Iterative feedback tuning of linear time-invariant mimo systems," in *Proceedings of the 37th IEEE Conference on Decision and Control (Cat. No. 98CH36171)*, vol. 4. IEEE, 1998, pp. 3893–3898.

[4] S. W. Sung and I.-B. Lee, "Limitations and countermeasures of pid controllers," *Industrial & engineering chemistry research*, vol. 35, no. 8, pp. 2596–2610, 1996.

[5] P. F. Odgaard, L. F. Larsen, R. Wisniewski, and T. G. Hovgaard, "On using pareto optimality to tune a linear model predictive controller for wind turbines," *Renewable Energy*, vol. 87, pp. 884–891, 2016.

[6] A. D. Ames and M. Powell, "Towards the unification of locomotion and manipulation through control lyapunov functions and quadratic programs," in *Control of Cyber-Physical Systems*. Springer, 2013, pp. 219–240.

[7] K. Galloway, K. Sreenath, A. D. Ames, and J. W. Grizzle, "Torque saturation in bipedal robotic walking through control lyapunov function-based quadratic programs," *IEEE Access*, vol. 3, pp. 323–332, 2015.

[8] J. Reher and A. D. Ames, "Control lyapunov functions for compliant hybrid zero dynamic walking," *ieee Transactions on Robotics and Automation*, In Preparation, 2021.

[9] E. Ambrose, W.-L. Ma, C. Hubicki, and A. D. Ames, "Toward benchmarking locomotion economy across design configurations on the modular robot: Amber-3m," in *2017 IEEE Conference on Control Technology and Applications (CCTA)*. IEEE, 2017, pp. 1270–1276.

[10] A. Robotics, https://www.agilityrobotics.com/robots#cassie, Last accessed on 2021-02-24.

[11] M. Birattari and J. Kacprzyk, *Tuning metaheuristics: a machine learning perspective*. Springer, 2009, vol. 197.

[12] M. Jun and M. G. Safonov, "Automatic pid tuning: An application of unfalsified control," in *Proceedings of the 1999 IEEE International Symposium on Computer Aided Control System Design (Cat. No. 99TH8404)*. IEEE, 1999, pp. 328–333.

[13] A. Marco, P. Hennig, J. Bohg, S. Schaal, and S. Trimpe, "Automatic lqr tuning based on gaussian process global optimization," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 270–277.

[14] M. Tucker, N. Csomay-Shanklin, W.-L. Ma, and A. D. Ames, "Preference-based learning for user-guided hzd gait generation on bipedal walking robots," 2020.

[15] J. W. Grizzle, C. Chevallereau, R. W. Sinnet, and A. D. Ames, "Models, feedback control, and open problems of 3d bipedal robotic walking," *Automatica*, vol. 50, no. 8, pp. 1955–1988, 2014.

[16] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris, *Feedback control of dynamic bipedal robot locomotion*. CRC press, 2018.

[17] A. Hereid and A. D. Ames, "FROST: Fast robot optimization and simulation toolkit," in *2017 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2017, pp. 719–726.

[18] J. Reher and A. D. Ames, "Inverse dynamics control of compliant hybrid zero dynamic walking," 2020.

[19] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.

[20] A. Isidori, *Nonlinear Control Systems, Third Edition*, ser. Communications and Control Engineering. Springer, 1995. [Online]. Available: https://doi.org/10.1007/978-1-84628-615-5

[21] J. Reher, C. Kann, and A. D. Ames, "An inverse dynamics approach to control lyapunov functions," 2020.

[22] Y. Sui, M. Zoghi, K. Hofmann, and Y. Yue, "Advancements in dueling bandits." in *IJCAI*, 2018, pp. 5502–5510.

[23] M. Tucker, M. Cheng, E. Novoseller, R. Cheng, Y. Yue, J. W. Burdick, and A. D. Ames, "Human preference-based learning for high-dimensional optimization of exoskeleton walking gaits," *arXiv preprint arXiv:2003.06495*, 2020.

[24] K. Li, M. Tucker, E. Bıyık, E. Novoseller, J. W. Burdick, Y. Sui, D. Sadigh, Y. Yue, and A. D. Ames, "Roial: Region of interest active learning for characterizing exoskeleton gait preference landscapes," *arXiv preprint arXiv:2011.04812*, 2020.

[25] W. Chu and Z. Ghahramani, "Preference learning with gaussian processes," in *Proceedings of the 22nd International Conference on Machine Learning*, ser. ICML '05. New York, NY, USA: Association for Computing Machinery, 2005, p. 137–144. [Online]. Available: https://doi.org/10.1145/1102351.1102369

[26] "Video of the experimental results." https://youtu.be/wrdNKK5JqJk.