

Neural Gaits: Learning Bipedal Locomotion via Control Barrier Functions and Zero Dynamics Policies

Ivan Dario Jimenez Rodriguez^{*†}

IVAN.JIEMENEZ@CALTECH.EDU

Noel Csomay-Shanklin^{*†}

NOELCS@CALTECH.EDU

Yisong Yue^{†‡}

YYUE@CALTECH.EDU

Aaron D. Ames[†]

AMES@CALTECH.EDU

[†]Caltech, Pasadena, CA, USA [‡]Argo AI

Editors: R. Firooz, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

This work presents Neural Gaits, a method for learning dynamic walking gaits through the enforcement of set invariance that can be refined episodically using experimental data from the robot. We frame walking as a set invariance problem enforceable via control barrier functions (CBFs) defined on the reduced-order dynamics quantifying the underactuated component of the robot: the zero dynamics. Our approach contains two learning modules: one for learning a policy that satisfies the CBF condition, and another for learning a residual dynamics model to refine imperfections of the nominal model. Importantly, learning only over the zero dynamics significantly reduces the dimensionality of the learning problem while using CBFs allows us to still make guarantees for the full-order system. The method is demonstrated experimentally on an underactuated bipedal robot, where we are able to show agile and dynamic locomotion, even with partially unknown dynamics.

Keywords: bipedal locomotion, zero dynamics, safety, robotics

1. Introduction

Realizing bipedal locomotion on legged robots is difficult due to the compounded complexity of nonlinear underactuated dynamics coupled with the hybrid nature of walking. Underactuation makes the application of classic nonlinear control approaches challenging, necessitating the use of offline optimization to generate periodic walking gaits. Due to the combinatorics of contact conditions resulting from the hybrid dynamics, feasibility of this optimization problem requires either fixing the contact times and positions (which can be vulnerable to perturbations) or expensive planning through the set of possible contact points. Pushing this offline optimization problem online allows for reactive controllers but requires the use of reduced-order models that limit formal guarantees. Despite impressive examples of implementations that deal with bipedal walking in practice, general bipedal locomotion with formal performance guarantees remains an open problem.

Prior Work in Control. In the control literature, bipedal locomotion follows two general branches: walking with guarantees of stability and predictive control approaches. Walking with guarantees usually relies on solving optimization programs offline to generate stable (periodic) gaits (Hereid and Ames, 2017). Above all, this approach relies on constraining walking to be a periodic orbit with assumed exponential stability on the underactuated coordinates of the robot. This underlying assumption can be problematic in safety critical settings when the gait must satisfy hard constraints such as staying on predetermined stepping stones (Csomay-Shanklin et al., 2021;

* These authors contributed equally to this work.

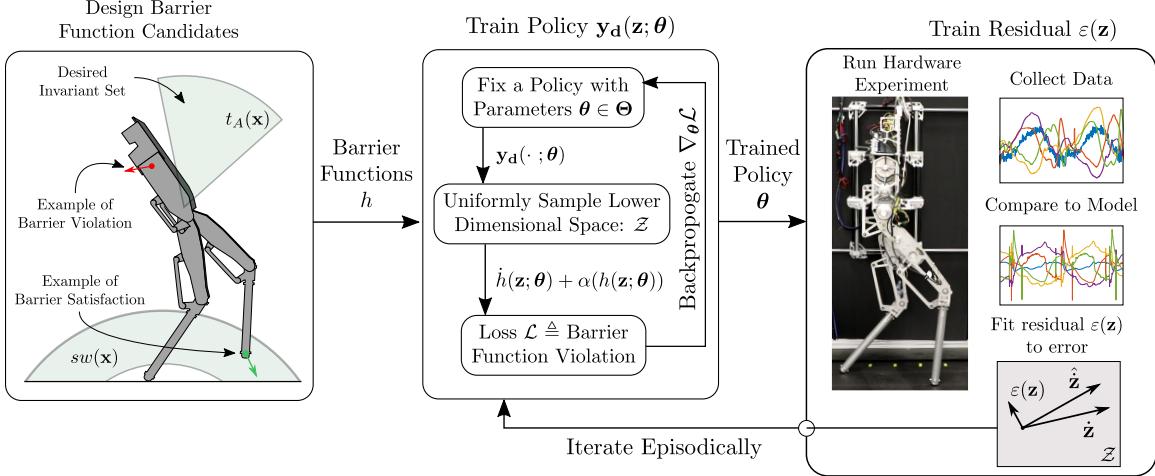


Figure 1: A depiction of the Neural Gaits framework. **Left:** Designing barrier function candidates that we use to formally describe walking. **Middle:** Training a policy capable of satisfying all the barrier condition in the zero dynamics state space where the constraint is enforced. **Right:** Collecting hardware data to train a residual zero dynamics model. We then refine the policy episodically using the augmented model.

Nguyen and Sreenath, 2015) and also precludes different walking modes such as period-two walking and, more generally, aperiodic locomotion (Xiong and Ames, 2019; Ames et al., 2017). This is particularly important given that disturbance rejection can require aperiodic behaviors (Raibert and Tello, 1986). Predictive control approaches on the other hand are able to avoid the aforementioned limitations by planning trajectories and/or policies online, and have shown great promise for quadrupedal robots (Di Carlo et al., 2018; Grandia et al., 2019). Their application to bipedal robots is comparatively sparse however, and has predominantly required static stability (Tedrake et al., 2015; Scianca et al., 2020) or simplified models to mitigate the computational complexity (Kuindersma et al., 2016; Apgar et al., 2018; Xiong and Ames, 2021). This leads to challenges when seeking formal guarantees for dynamic bipedal locomotion in the presence of model mismatch between the planning and low-level control layers.

Prior Work in Learning. Prior work in machine learning has produced impressive results towards realizing legged locomotion using reinforcement learning (Lee et al., 2020; Siekmann et al., 2020; Castillo et al., 2021; Heess et al., 2017). These methods use relatively simple reward functions along with sophisticated simulations to generate large amounts of data to train a policy capable of traversing a variety of terrains. Still, these algorithms can be fragile when facing environments outside of the training dataset and are data inefficient due to not exploiting the full dynamics structure. These challenges make it difficult to reliably apply these methods on complex hardware systems. Other works have attempted to use reinforcement learning to train a parameterization of a Control Barrier Function (CBF) Control Lyapunov Function (CLF) Quadratic Program Controller (CBF CLF QP) (Choi et al., 2020; Csomay-Shanklin et al., 2021). Differing from these works, this paper does not learn a projection of the modeling error onto the CLF and CBF constraints; instead we learn the projection of our modeling error on the zero dynamics. Furthermore, we do not specify a desired trajectory but rather provide a set of barriers that imply walking as emergent behavior.

Our Contributions. In this work we are instead interested in automatically discovering good walking policies by integrating learning with control-theoretic formulations of stable walking. Rather than optimizing a simple reward/cost function, our policies learn to satisfy algebraic forward set in-

variance conditions, certified by control barrier functions, that characterize good walking behavior (as illustrated in the first block of Figure 1). This integrated approach offers three potential benefits. First, we are able to certifiably handle impact uncertainty by training policies that satisfy conditions for good impact behavior on a region surrounding the nominal impact guard. Second, our sampling-based approach is focused on a lower-dimensional state space which can be more data efficient and, under the assumption that walking is a form of set invariance, can produce policies with performance certificates. The latter point contrasts with offline or predictive trajectory generation perspectives where, even if you enforce control theoretic conditions over the predicted trajectory, you cannot certify performance over a set surrounding the trajectory, thereby breaking set invariance guarantees even under small perturbations. Furthermore, leveraging zero dynamics significantly reduces the dimensionality of the learning problem while remaining compatible with the use of control barrier functions, thus enabling guarantees for the full-order system (see Section 3.3). Third, rather than relying on the artificial constraint of periodic orbits, we are able to characterize walking solely as set invariance while retaining stability guarantees. The combined benefit is a reliable and efficient approach for designing gaits that can be deployed on hardware platforms.

Our proposed approach, called *Neural Gaits*, is composed of two learning modules. The first module trains a policy (which generates walking gaits) to minimize the violations of forward set invariance, implemented using control-theoretic barrier conditions as shown in the second block of Figure 1. In doing so, we can guarantee forward set invariance, which implies (under suitable assumptions) indefinite stable walking. The second module trains a residual dynamics model to refine imperfections of the current dynamics model based on hardware experiments as represented by the arrow between block three and two in Figure 1. Both modules are then iterated on episodically with hardware experiments in the loop. We also build upon recent work in training ODE-based systems (of which locomotive walking is an instance of), such as LyaNet (Rodriguez et al., 2022) and Neural ODEs (Chen et al., 2018), in order to develop an effective training approach.

We empirically demonstrate our approach on the AMBER-3M hardware platform (Ambrose et al., 2017) with partially unknown dynamics. We show that the resulting policy is capable of making the robot walk under significant model mismatch and adapts to improve barrier satisfaction across episodes. To the best of our knowledge, this is the first successful demonstration of integrated learning and control for bipedal locomotion with stability guarantees.

2. Preliminaries

We provide a brief introduction of zero dynamics, hybrid dynamical systems, and control barrier functions, which are necessary fundamentals to understand the proposed formulation in Section 3.

2.1. Output and Zero Dynamics

Consider the general nonlinear ordinary differential equation:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, \mathbf{u}), \quad \mathbf{x}(0) = \mathbf{x}_0, \quad (1)$$

with states $\mathbf{x} \in \mathcal{X} \subseteq \mathbb{R}^n$, inputs $\mathbf{u} \in \mathcal{U} \subseteq \mathbb{R}^m$, and dynamics $\mathbf{f} : \mathcal{X} \times \mathcal{U} \rightarrow \mathbb{R}^n$ with \mathbf{f} locally Lipschitz in both arguments. For mechanical systems, we specialize to the control-affine case:

$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}, \quad (2)$$

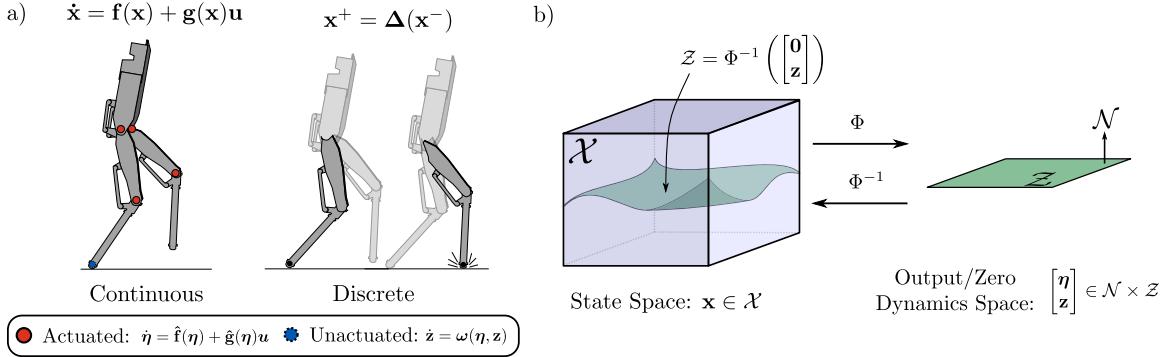


Figure 2: a) The continuous and discrete phases of the robot, with actuated (η) and unactuated (\mathbf{z}) coordinates depicted. b) The diffeomorphism Φ , and the relationship between state space coordinates and output/zero dynamics coordinates.

where $\mathbf{f} : \mathcal{X} \rightarrow \mathbb{R}^n$ and $\mathbf{g} : \mathcal{X} \rightarrow \mathbb{R}^{n \times m}$ are assumed to be locally Lipschitz. Denoting a parameter in a parameter space $\boldsymbol{\theta} \in \Theta$, we can define a collection of k outputs $\mathbf{y} : \mathcal{X} \times \Theta \rightarrow \mathbb{R}^k$ parameterized by $\boldsymbol{\theta}$ that we would like to converge to zero as:

$$\mathbf{y}(\mathbf{x}; \boldsymbol{\theta}) = \mathbf{y}_a(\mathbf{x}) - \mathbf{y}_d(\mathbf{x}; \boldsymbol{\theta}), \quad (3)$$

where $\mathbf{y}_a : \mathcal{X} \rightarrow \mathbb{R}^k$ are the measured outputs, and $\mathbf{y}_d : \mathcal{X} \times \Theta \rightarrow \mathbb{R}^k$ are the desired outputs. For locomotion, the outputs are typically taken either as joint angles (as done in this work), or as center of mass and foot positions. For the policy y_d learned in this work and shown in the center block of Figure 1, $\boldsymbol{\theta}$ corresponds to neural network parameters. Although all the concepts may be extended to systems with valid decomposition into output and zero dynamics coordinates (which includes all mechanical systems), for simplicity the remainder of the exposition will be restricted to the setting used in this work, namely with $k = 4$ and \mathbf{y}_a taken to be the actuated joint angles of the robot. For a complete description of output coordinates and zero dynamics, we refer to (Isidori, 1995).

Given these outputs \mathbf{y} , we can separate the actuated and the unactuated coordinates for the robot, which are shown in Figure 2a. As these outputs are *vector relative degree 2*, we can define error coordinates $\boldsymbol{\eta}_i : \mathcal{X} \rightarrow \mathcal{N}_i \subseteq \mathbb{R}^2$ for $i = 1, \dots, 4$ as $\boldsymbol{\eta}_i = [y_i^\top, \dot{y}_i^\top]^\top$, as well as the collection of errors $\boldsymbol{\eta} = [\boldsymbol{\eta}_1^\top, \dots, \boldsymbol{\eta}_4^\top]^\top$. Then, there exist 2 linearly independent functions $z_i : \mathcal{X} \rightarrow \mathcal{Z}_i \subseteq \mathbb{R}$ for $i = 1, 2$ such that $\nabla_{\mathbf{x}} z_i(\mathbf{x}) g(\mathbf{x}) \equiv 0$, and $\nabla_{\mathbf{x}} z_i(\mathbf{x})$ is linearly independent from $\nabla_{\mathbf{x}} \boldsymbol{\eta}_{i,j}(\mathbf{x})$ for $i = 1, \dots, 4$ and $j = 1, 2$. We can then construct a diffeomorphism¹ $\Phi : \mathcal{X} \times \Theta \rightarrow \mathcal{N} \times \mathcal{Z}$:

$$\begin{bmatrix} \boldsymbol{\eta} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \Phi_\eta(\mathbf{x}; \boldsymbol{\theta}) \\ \Phi_\mathbf{z}(\mathbf{x}) \end{bmatrix} \triangleq \Phi(\mathbf{x}; \boldsymbol{\theta}), \quad \mathbf{x} = \Phi^{-1} \left(\begin{bmatrix} \boldsymbol{\eta} \\ \mathbf{z} \end{bmatrix}; \boldsymbol{\theta} \right),$$

as shown in Figure 2b. Under this coordinate transformation, the system dynamics become:

$$\begin{bmatrix} \dot{\boldsymbol{\eta}} \\ \dot{\mathbf{z}} \end{bmatrix} = \begin{bmatrix} \hat{\mathbf{f}}(\boldsymbol{\eta}; \boldsymbol{\theta}) + \hat{\mathbf{g}}(\boldsymbol{\eta}; \boldsymbol{\theta})\mathbf{u} \\ \omega(\boldsymbol{\eta}, \mathbf{z}; \boldsymbol{\theta}) \end{bmatrix}, \quad (4)$$

where $\hat{\mathbf{f}}$, $\hat{\mathbf{g}}$ and ω are the projection of the dynamics through the diffeomorphism Φ .

The *zero dynamics manifold* $\mathcal{Z} \subset \mathcal{X}$ is thus the space where errors have been driven to zero:

$$\mathcal{Z} = \{\mathbf{x} \in \mathcal{X} : \boldsymbol{\eta}(\mathbf{x}) = 0\},$$

1. This is differentiable in the first argument and differentiable almost everywhere in the second argument.

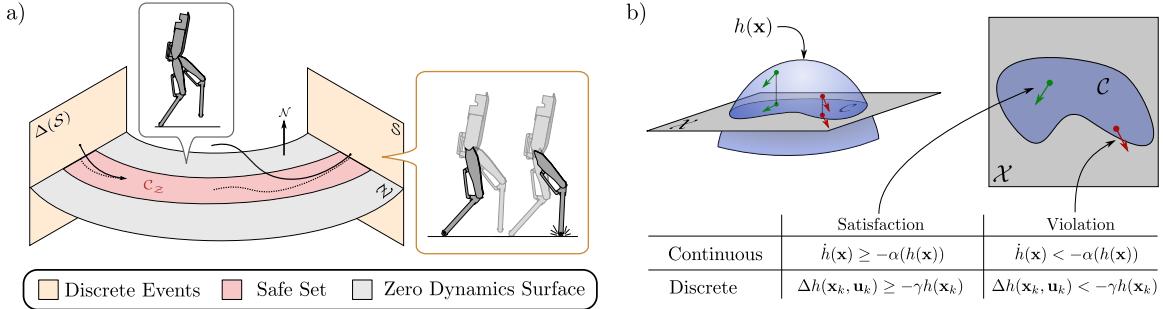


Figure 3: **a)** The guard \mathcal{S} , reset map $\Delta(\mathcal{S})$, and safe set $\mathcal{C}_{\mathcal{Z}}$ are visualized in the state space decomposed into output \mathcal{N} and zero dynamics \mathcal{Z} coordinates. **b)** A safe set defined as regions of the state space (gray square) where h is positive (blue region). Satisfying the CBF condition implies that the flows/discrete updates of the system will not leave the safe set (although may approach the boundary). Violations imply a flow that could potentially leave the safe set.

as seen in Figure 2a. Observe that for $\mathbf{z} \in \mathcal{Z}$, we have that $\dot{\mathbf{z}} = \omega(\mathbf{0}, \mathbf{z}; \theta)$. The power of the method of zero dynamics lies in that it allows for guarantees about the full nonlinear dynamics by considering only a subspace of significantly smaller dimensionality (Isidori, 1995). Notice that although the input does not appear in the zero dynamics ω in (4), the parameters of the policy θ do. This realization motivates the use of the policy as a way to influence the zero dynamics and enforce the desired barrier functions, as introduced below. Finally, in this work we will learn residual dynamics $\varepsilon(\mathbf{z})$ on the zero dynamics manifold for a corrected zero dynamics $\dot{\mathbf{z}} = \omega(\mathbf{0}, \mathbf{z}) + \varepsilon(\mathbf{z})$ that compensate for modeling error. This process is captured in the episodic iteration of Figure 1.

2.2. Hybrid Dynamics

Walking consists of continuous evolution with discrete impacts occurring as contact is made and broken (e.g., the feet with the ground). This sequence of continuous and discrete dynamics is shown in Figure 3a can be modeled in the language of *hybrid systems* as:

$$\mathcal{HC} = \begin{cases} \dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u} & \mathbf{x} \in \mathcal{D} \setminus \mathcal{S} \\ \mathbf{x}^+ = \Delta(\mathbf{x}^-) & \mathbf{x} \in \mathcal{S} \subset \mathcal{D}, \end{cases}$$

where $\mathcal{D} \subset \mathcal{X}$ is the domain where $\mathbf{x}(t)$ evolves. The *guard*, $\mathcal{S} \subset \mathcal{X}$, corresponds to the set of states where the foot comes in contact with the floor. The *reset map*, $\Delta : \mathcal{S} \rightarrow \mathcal{D}$ models the instantaneous sign flip of velocities observed when two rigid bodies collide (the foot with the ground). Furthermore, \mathcal{HC} can be projected through the diffeomorphism Φ to exploit the decomposition into output and zero dynamics. For more details, we refer to (Westervelt et al., 2018).

2.3. Control Barrier Function Certificates

Barrier function certificates allow us to make the notion of *safety* rigorous in the context of the dynamical system in Equation (1). We begin by specifying a set that we wish to render safe:

$$\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) \geq 0\} \subset \mathcal{X}, \quad (5)$$

where $h : \mathcal{X} \rightarrow \mathbb{R}$ is a continuously differentiable function. In the case of bipedal walking, safe sets can describe conditions such as admissible torso angles and reasonable foot placement as shown in

the first block of Figure 1. We assume that the compact set \mathcal{C} is nonempty, has a non-empty interior, and does not contain isolated fixed-points. We say that \mathcal{C} is *safe* or *forward invariant* if $x(t_0) \in \mathcal{C}$ implies that $x(t) \in \mathcal{C}$ for all $t \geq t_0$. With this, we have the following condition for safety (see (Ames et al., 2019) for a brief history of this approach):

Definition 1 (Control Barrier Function (CBF), (Ames et al., 2016)) Consider \mathcal{C} as defined in Equation (5) where the continuously differentiable function h has nonvanishing gradients $Dh(\mathbf{x}) \neq 0$ for all \mathbf{x} in the boundary of \mathcal{C} defined as $\partial\mathcal{C} = \{\mathbf{x} \in \mathcal{X} : h(\mathbf{x}) = 0\}$. The function h is a Control Barrier Function (CBF) for Equation (1) on \mathcal{C} if there exists $\alpha \in \mathcal{K}_{\infty,e}$ such that for all $\mathbf{x} \in \mathcal{C}$:

$$\dot{h}(\mathbf{x}) = \frac{\partial h}{\partial \mathbf{x}}(\mathbf{x}) \mathbf{f}(\mathbf{x}, \mathbf{u}) \geq -\alpha(h(\mathbf{x})). \quad (6)$$

A function α is in the family of class- $\mathcal{K}_{\infty,e}$ functions if for all $a < b$, $\alpha(a) < \alpha(b)$, $\alpha(0) = 0$, $\lim_{a \rightarrow \infty} \alpha(a) = \infty$ and $\lim_{a \rightarrow -\infty} \alpha(a) = -\infty$. In defining the CBF, we can parameterize the set of all feedback controllers guaranteeing safety as:

$$K_{cbf}(\mathbf{x}) = \left\{ \mathbf{u} \in \mathcal{U} : \dot{h}(\mathbf{x}, \mathbf{u}) \geq -\alpha(h(\mathbf{x})) \right\}. \quad (7)$$

Similarly, this notion can be extended to discrete-time dynamical systems via:

$$\Delta h(\mathbf{x}_k, \mathbf{u}_k) \triangleq h(\mathbf{x}_{k+1}) - h(\mathbf{x}_k) \geq -\gamma h(\mathbf{x}_k), \quad 0 < \gamma \leq 1, \quad (8)$$

as seen in Figure 3b. This leads to the following necessary and sufficient condition for safety:

Theorem 2 (Control Barrier Function Certificates, (Ames et al., 2016)) Given a feedback controller $\mathbf{u} = k(\mathbf{x})$, the set \mathcal{C} is safe if and only if $\mathbf{u}(\mathbf{x}) \in K_{cbf}(\mathbf{x})$.

3. Neural Gaits: Locomotion as a Barrier Satisfiability Problem

We now present our Neural Gaits approach, as depicted in Figure 1. Instead of taking a controller-design perspective, we will take one of reference trajectory design – specifically, we fix a controller structure $\mathbf{u}(\mathbf{x}; \theta)$, which is parameterized by θ through the definition of $\mathbf{y}(\mathbf{x}; \theta)$. Our method relies on the assumption that good walking can be characterized as a forward invariant set. Thus, the first step of the method requires us to define a set of barrier functions that imply good walking. In the following discussion, we will only consider barrier functions defined on the zero dynamics surface, i.e. $h : \mathcal{Z} \rightarrow \mathbb{R}$ as defined when the error coordinates are zero ($\eta = 0$). Importantly, the guarantees made on $\mathcal{Z} \subset \mathcal{X}$ have relevance to the full state space, as is made precise in Section 3.4.

After constructing a collection of barrier functions, we train a policy \mathbf{y}_d that ensures the system stays safe by minimizing the violation of the barrier function conditions (6) and (8) over regions of the state space. The resulting policy renders the intersection of the safe set for all barriers forward invariant. Finally, to mitigate model mismatch, we train a residual term $\varepsilon(\mathbf{z})$ on the zero dynamics. These corrected zero dynamics are then used to refine the existing policy episodically until the desired walking performance is achieved.

3.1. Learning the Policy \mathbf{y}_d

Our learning approach builds upon and unifies two lines of work. The first studies how to characterize good walking behavior as set invariance via a collection of barrier function candidates (Ames et al., 2017). The second studies how to train neural ODEs to satisfy control-theoretic properties such as Lyapunov stability (Rodriguez et al., 2022), which we extend to the barrier setting.

Learning in the Zero Dynamics. Recall from Section 2.1 that the error and zero dynamics coordinates are computed from the states using the diffeomorphism Φ , which only depends on the policy through Φ_η . We thus parameterize the policy as a function of the projection of the state onto the zero dynamics manifold and parameters θ , i.e. $\mathbf{y}_d(\mathbf{x}) = \mathbf{y}_d(\Phi_{\mathbf{z}}(\mathbf{x}); \theta)$. In other words, \mathbf{y}_d only depends on the unactuated degrees of freedom of the system (e.g., the unactuated joint in Figure 2) rather than the full state. Therefore, when there is no error (i.e. $\eta = 0$) we have that $\mathbf{z} \in \mathcal{Z}$ with dynamics $\dot{\mathbf{z}} = \omega(0, \mathbf{z}; \theta)$. Note, importantly, that even when the error coordinates are zero, the zero dynamics are still a function of \mathbf{y}_d and therefore θ . This implies that the zero dynamics are influenced by the parameters of the policy even though the control inputs are not present in ω .

Learning to Satisfy Barrier Conditions. Taking inspiration from (Hsu et al., 2015) and (Ames et al., 2017), we assume that walking can be characterized as set invariance via a collection of barrier function candidates $\mathcal{H} = \{h_i\}_{i=1}^N$ (see Table 1 discussed later in Section 3.2). To each h_i , we associate a *region at risk* $\bar{\mathcal{S}}_i \subseteq \mathcal{Z}$ where the barrier function is enforced. We define a set of neural network parameters that render the region at risk safe under the barrier definition:

$$\Theta_i = \{\theta \in \Theta : \forall_{\mathbf{z} \in \bar{\mathcal{S}}_i} \dot{h}_i(\mathbf{z}; \theta) \geq -\alpha(h_i(\mathbf{z}; \theta))\}. \quad (9)$$

In other words, each Θ_i corresponds to the set of policy parameters that render the set $\bar{\mathcal{S}}_i$ safe.

Thus, our learning problem is equivalent to finding a set of parameters $\theta \in \bigcap_{i=1}^N \Theta_i$ that render the system safe in all regions at risk. Similar to the Lyapunov Loss studied in (Rodriguez et al., 2022), we introduce the concept of *Barrier Loss* as a learning signal for training:

Definition 3 (Barrier Loss) For a set of barrier function candidates $\mathcal{H} = \{h_i\}_{i=1}^N$ and corresponding regions at risk $\bar{\mathcal{S}}_i \subset \mathcal{Z}$ on the zero dynamics, a Barrier Loss, $\mathcal{L} : \Theta \rightarrow \mathbb{R}_{\geq 0}$, is defined as:

$$\mathcal{L}(\theta) = \sum_{i=1}^N \int_{\bar{\mathcal{S}}_i} \max\{0, -\dot{h}_i(\mathbf{z}; \theta) - \alpha(h_i(\mathbf{z}; \theta))\} d\mathbf{z}. \quad (10)$$

When a choice of parameters θ achieves zero Barrier Loss, then the safety of the zero dynamics is guaranteed by satisfying the forward invariance condition of the barrier functions:

Theorem 4 (Zero Barrier Loss Implies Safety of Zero Dynamics) The zero dynamics is guaranteed to be safe in all its regions at risk if and only if we find a θ^* that attains $\mathcal{L}(\theta^*) = 0$.

Proof Notice that for all $i \in \{1 \dots N\}$ both \dot{h}_i and $\alpha \circ h_i$ are continuous functions. This implies that for all $\mathbf{z} \in \mathcal{Z}$ and $\theta \in \Theta$, $\max\{0, -\dot{h}_i(\mathbf{z}; \theta) - \alpha(h_i(\mathbf{z}; \theta))\}$ is a continuous non-negative real function. It is well known that a continuous non-negative real function will have zero integral if and only if it is the zero function. We specialize this statement for the terms in our loss as follows:

$$\forall_{\mathbf{z} \in \bar{\mathcal{S}}_i} \max\{0, -\dot{h}_i(\mathbf{z}; \theta) - \alpha(h_i(\mathbf{z}; \theta))\} = 0 \Leftrightarrow \int_{\bar{\mathcal{S}}_i} \max\{0, -\dot{h}_i(\mathbf{z}; \theta) - \alpha(h_i(\mathbf{z}; \theta))\} d\mathbf{z} = 0 \quad (11)$$

It is clear that the sum in $\mathcal{L}(\theta)$ will be zero if and only if each integral term is zero since each integral is a non-negative function. Thus we can conclude that $\mathcal{L}(\theta^*) = 0$ if and only if

$$\forall_{i \in \{1 \dots N\}, \mathbf{z} \in \bar{\mathcal{S}}_i} \max\{0, -\dot{h}_i(\mathbf{z}; \theta^*) - \alpha(h_i(\mathbf{z}; \theta^*))\} = 0.$$

For any barrier h_i and $\mathbf{z} \in \mathcal{Z}$ you can see that $\max\{0, -\dot{h}_i(\mathbf{z}; \theta^*) - \alpha(h_i(\mathbf{z}; \theta^*))\} = 0$ implies that:

$$-\dot{h}_i(\mathbf{z}; \theta^*) - \alpha(h_i(\mathbf{z}; \theta^*)) \leq 0 \implies \dot{h}_i(\mathbf{z}; \theta^*) \geq -\alpha(h_i(\mathbf{z}; \theta^*)),$$

i.e. the safety condition for the barrier is satisfied. ■

3.2. Instantiation for Bipedal Walking

Table 1 describes the barrier functions used in our experiments, which take inspiration from (Ames et al., 2017). We depict some of these conditions on the robot in Figure 4a. As all barrier functions $h_i : \mathcal{X} \rightarrow \mathbb{R}$ are enforced on the zero dynamics surface, we will write them implicitly as $h_i \circ \Phi^{-1}(\mathbf{0}, \cdot; \boldsymbol{\theta}) : \mathcal{Z} \rightarrow \mathbb{R}$ for $i \in \{1 \dots N\}$ with $N = 5$ in this instantiation. In Table 1, t_A represents the torso angle, and p_x and p_z represent the x and z position of the swing foot, respectively. In addition to continuous time conditions, various conditions needed to be enforced on the guard, namely enforcing the location of the guard, symmetry of the model before and after impact, and a guard mapping condition. Interestingly, although these barriers would be relative degree two on the full state dynamics, they are directly enforceable as relative degree one barriers on the zero dynamics. This can be seen by treating \mathbf{y}_d as the input to the zero dynamics, and observing that the zero dynamics themselves are functions of \mathbf{y}_d .

Note that these barrier functions h are defined over the space \mathcal{Z} , as, given a policy $\mathbf{y}_d(\cdot; \boldsymbol{\theta}) : \mathcal{Z} \rightarrow \mathbb{R}^4$, the mapping $\Phi^{-1} : \mathcal{N} \times \mathcal{Z} \rightarrow \mathcal{X}$ is uniquely defined. We take inspiration from reduced order models, and specifically the notion of orbital energy (Pratt and Drakunov, 2007) to define a set $\mathcal{Z}_{\mathcal{O}} \subset \mathcal{Z}$ with reasonably bounded orbital energies as our first region at risk. We also define the set $\mathcal{S}_{\epsilon} \subset \mathcal{Z}_{\mathcal{O}}$ which contains the part of the guard in $\mathcal{Z}_{\mathcal{O}}$ as well as a small region around it where discrete-time guard conditions are enforced. We learn policies that satisfy the barrier conditions on these regions of the zero dynamics by penalizing the violation of the constraints shown in Table 1. Notice that penalizing guard constraints over a region results in policies that are robust to impact modeling error since the policy must be prepared to change stance foot at any point in \mathcal{S}_{ϵ} rather than just the guard \mathcal{S} .

Learning Optimization Details. Evaluating the Barrier Loss in Equation (10) requires solving an integral that is in general intractable. Instead, we use Monte Carlo sampling to approximate the integral. Since our approach follows Algorithm 1 of (Rodriguez et al., 2022) we refer to it for more details while noting that we optimize for the Barrier Loss rather than the Lyapunov Loss. A key ingredient in the Monte Carlo sampling approach in (Rodriguez et al., 2022) is defining a compact support set to sample from (i.e., where the barrier condition should be satisfied). In our work this compact support set directly corresponds to the region at risk for each barrier condition.

3.3. Learning the Residual Zero Dynamics $\varepsilon(\mathbf{z})$

As outlined in Figure 1, we can improve upon the nominal zero dynamics model by collecting trajectories of the robot executing the resulting policy in hardware. We can then use those trajectories to learn a residual error term on the zero dynamics $\hat{\mathbf{z}} = \omega(0, \mathbf{z}; \boldsymbol{\theta}) + \varepsilon(\mathbf{z})$ where ε is the learned

Torso Angle	$\{\mathbf{z} \in \mathcal{Z}_{\mathcal{O}}\}$	$-\frac{\pi}{10} \leq \theta_t(\mathbf{z}) \leq 0.05$
Swing Foot Clearance	$\{\mathbf{z} \in \mathcal{Z}_{\mathcal{O}}\}$	$0 \leq (p_x(\mathbf{z}) - c_x)^2 + (p_z(\mathbf{z}) - c_z)^2 - r^2 \leq 0.3$
Impact Mapping	$\{\mathbf{z} \in \mathcal{S}_{\epsilon}\}$	$-0.15 \leq \Delta(\mathbf{z}) + \mathbf{z} \leq 0.15$
Symmetry	$\{\mathbf{z} \in \mathcal{S}_{\epsilon}\}$	$\mathbf{y}(\mathbf{z}) = \mathbf{y}(\Delta(\mathbf{z}))$
Foot on Guard	$\{\mathbf{z} \in \mathcal{S}_{\epsilon}\}$	$p_z(\mathbf{z}) = 0$

Table 1: Barrier functions used to characterize bipedal walking, and the associated regions at risk in which they are enforced. The first two are enforced over the continuous dynamics, and the bottom three in a buffered region of the guard. The strict equality on symmetry and the foot on guard conditions were also enforced as a training loss.

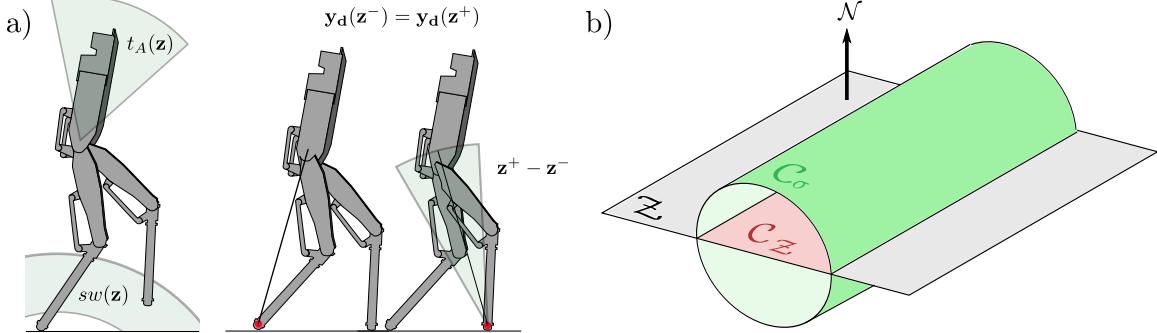


Figure 4: a) A depiction of the barrier functions used to enforce walking as set invariance. On the left are the two continuous time barrier conditions, and on the right the three barrier conditions enforced at the guard. The red dot on the foot indicates the stance foot, and b) The safe set on the zero dynamics \mathcal{C}_Z , as certified by the proposed learning method, and the combined safe set \mathcal{C}_σ , and described in Theorem 5.

residual term. We model this residual term using Neural ODEs (Chen et al., 2018), which are naturally compatible with our policy learning approach. We can iterate this process multiple times, alternating between learning θ and ε until the resulting policy achieves the desired behavior.

3.4. Providing Guarantees in the Full State Space

Assuming a controller which exponentially converges the outputs $\mathbf{y}(\mathbf{x})$, for example a feedback linearizing or control Lyapunov function based controller, the converse Lyapunov theorem allows us to construct a Lyapunov function $V_\eta : \mathcal{N} \rightarrow \mathbb{R}$ verifying the exponential convergence of the outputs. Along with a certificate of safety $h_Z : \mathcal{Z} \rightarrow \mathbb{R}$ on the zero dynamics space, we can construct a set in the combined space $\mathcal{N} \times \mathcal{Z}$ which is safe, and has a barrier function certificate. This is described in the following theorem.

Theorem 5 *Let $V_\eta = \eta^\top \mathbf{P} \eta : \mathcal{N} \rightarrow \mathbb{R}$ be an exponential control Lyapunov function for the output dynamics with $\dot{V}_\eta \leq -\gamma V_\eta$ and $h_Z : \mathcal{Z} \rightarrow \mathbb{R}$ be a barrier function on the zero dynamics with safe set \mathcal{C}_Z . Then, there exists a constant $\sigma \geq 0$ and $c \geq 0$ such that if $\dot{h}_Z(z) \geq -\alpha h_Z(z) + c$ with $\alpha \leq \frac{\gamma}{2}$, the barrier function $h(\eta, \mathbf{z}) = h_Z(\mathbf{z}) - \sigma V_\eta(\eta)$ is safe with set \mathcal{C}_σ .*

Proof First note that the derivative of the function is given by:

$$\begin{aligned} \dot{h} &= \frac{\partial h_Z}{\partial \mathbf{z}}(\mathbf{z}) \mathbf{w}(\eta, \mathbf{z}) - \sigma \dot{V}_\eta(\eta) \\ &\geq -\alpha h_Z(\mathbf{z}) + c - \left| \frac{\partial h_Z}{\partial \mathbf{z}}(\mathbf{z}) (\mathbf{w}(\eta, \mathbf{z}) - \mathbf{w}(0, \mathbf{z})) \right| + \sigma \gamma V_\eta(\eta) \\ &\geq -\alpha h(\eta, \mathbf{z}) + c - L_{h_Z} L_{\omega_\eta} \|\eta\|_2 + \frac{\sigma \gamma}{2} \lambda_{\min}(\mathbf{P}) \|\eta\|_2^2, \end{aligned} \quad (12)$$

where the third line follows from Cauchy Schwartz, the fact that h_Z and $\omega(\eta, \mathbf{z})$ are locally Lipschitz with Lipschitz constants L_{h_Z} and L_{ω_η} , respectively, converse Lyapunov, and the assumption that $\alpha \leq \frac{\gamma}{2}$. Taking $\beta_1 = L_{h_Z} L_{\omega_\eta}$, and $\beta_2 = \frac{\gamma}{2} \lambda_{\min}(\mathbf{P})$, we observe that $-\beta_1 \|\eta\|_2 + \sigma \beta_2 \|\eta\|_2^2 \geq -\frac{\beta_2^2}{4\sigma\beta_3} \triangleq c$. By taking c defined as such, we achieve the desired result. ■

The above theorem motivates the perspective of this work: satisfying barrier function certificates in the zero dynamics enables reasoning about safe sets in the complete state space. Note that the hybrid case is not addressed here, and is an interesting direction for future theoretical work.

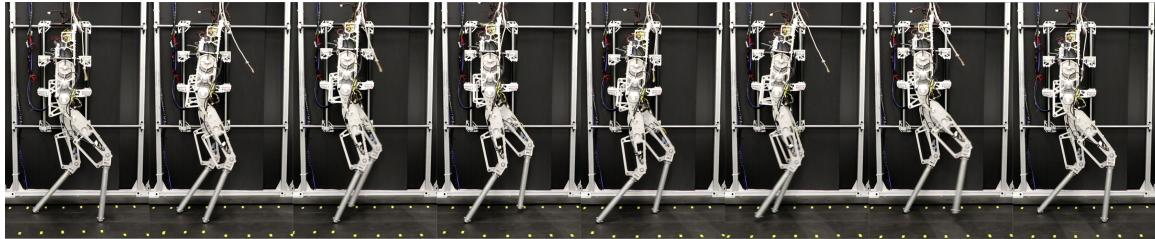


Figure 5: Gait tiles of the neural network encoding the final trained policy running in real time on the AMBER-3M robot. For a video discussing the methodology and summarizing the hardware results, please refer to ([vid](#)).

4. Simulation and Experimental Results

The hardware platform used in this work was the planar underactuated biped AMBER-3M ([Ambrose et al., 2017](#)), which has actuators on the hips and knees, and point contact feet. Both in simulation, where the RaiSim ([Hwangbo et al., 2018](#)) environment was used, and on hardware, the pipeline went as follows: the zero dynamics coordinate \mathbf{z} was estimated, the Neural Network policy $\mathbf{y}_d(\mathbf{z}; \theta)$ was evaluated, and the desired output values were passed to a PD controller running at 1kHz. The policy $\mathbf{y}_d(\mathbf{z}; \theta)$ was randomly initialized and was trained for 1000 epochs. The AdamW optimizer was used in PyTorch ([Paszke et al., 2019](#)) with an initial learning rate of 10^{-2} , weight decay of 10^{-4} , with a learning rate decay schedule at epochs 100, 400, and 800. Initially, the "gait" had the robots leg flailing randomly in the air, and when integrated resulted in the robot falling over. Once the loss converged, the policy had a loss in the order of 5×10^{-3} , and was able to walk stably in the simulation. The neural network ran in closed loop on the hardware platform and was called at approximately 500 Hz to produce desired outputs for the system to track. Unlike simulation, once tested on hardware, the policy resulted in the robot stumbling forward, unable to walk without falling. Data was collected over various trials, after which the methodology proposed in Section 3.3 was used to learn the residual of the model uncertainty, as projected to the zero dynamics space. During this process, Adam and other SGD methods were numerically unstable even with gradient clipping, so Nero ([Liu et al., 2021](#)) was used instead.

Once a residual term was learned, a new policy $\mathbf{y}_d(\mathbf{z}, \theta)$ was trained with the updated dynamics (warm started with the policy from the previous iteration). After convergence, the gait was again tried on hardware. The gait was significantly more stable, and able to walk without assistance; however, the gait was not robust to walking speeds. Therefore, the process was repeated, and again a new policy was learned. When testing that policy, the robot was able to walk on its own, and was robust to different walking speeds. A sample gait is shown on Figure 5. The complete code can be found here ([git](#)).

5. Conclusion

In this work, barrier functions, machine learning, and dimension reduction via zero dynamics were combined to provide a novel way of generating walking behaviors for a bipedal robot. Our approach used learning in two places: policy design and residual dynamics modeling via data collection on hardware. The proposed method culminated in a demonstration of agile and robust locomotion on hardware. Future work includes studying more complex robots, online learning, as well as policy learning for new behaviors (e.g., walking up stairs).

6. Acknowledgements

The authors would like to thank Min Dai, Ryan Cosner, and Andrew Taylor for their insightful discussions related to walking, barrier functions, and projection to state safety.

References

- Learning Code. URL <https://github.com/ivandariojr/NeuralGaits>.
- Supplementary Video. URL <https://www.youtube.com/watch?v=8TeXd0AYtpA>.
- Eric Ambrose, Wen-Loong Ma, Christian Hubicki, and Aaron 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)*, pages 1270–1276, Mauna Lani Resort, HI, USA, August 2017. IEEE. ISBN 978-1-5090-2182-6. doi: 10.1109/CCTA.2017.8062633. URL <http://ieeexplore.ieee.org/document/8062633/>.
- Aaron D Ames, Xiangru Xu, Jessy W Grizzle, and Paulo Tabuada. Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8):3861–3876, 2016.
- Aaron D. Ames, Paulo Tabuada, Austin Jones, Wen-Loong Ma, Matthias Rungger, Bastian Schürmann, Shishir Kolathaya, and Jessy W. Grizzle. First steps toward formal controller synthesis for bipedal robots with experimental implementation. *Nonlinear Analysis: Hybrid Systems*, 25:155–173, August 2017. ISSN 1751570X. doi: 10.1016/j.nahs.2017.01.002. URL <https://linkinghub.elsevier.com/retrieve/pii/S1751570X1730002X>.
- Aaron D Ames, Samuel Coogan, Magnus Egerstedt, Gennaro Notomista, Koushil Sreenath, and Paulo Tabuada. Control barrier functions: Theory and applications. In *2019 18th European control conference (ECC)*, pages 3420–3431. IEEE, 2019.
- Taylor Apgar, Patrick Clary, Kevin Green, Alan Fern, and Jonathan W Hurst. Fast online trajectory optimization for the bipedal robot cassie. In *Robotics: Science and Systems*, volume 101, page 14, 2018.
- Guillermo A. Castillo, Bowen Weng, Wei Zhang, and Ayonga Hereid. Robust feedback motion policy design using reinforcement learning on a 3d digit bipedal robot. *CoRR*, abs/2103.15309, 2021. URL <https://arxiv.org/abs/2103.15309>.
- Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.
- Jason Choi, Fernando Castaneda, Claire J Tomlin, and Koushil Sreenath. Reinforcement learning for safety-critical control under model uncertainty, using control lyapunov functions and control barrier functions. *arXiv preprint arXiv:2004.07584*, 2020.
- Noel Csomay-Shanklin, Ryan K Cosner, Min Dai, Andrew J Taylor, and Aaron D Ames. Episodic learning for safe bipedal locomotion with control barrier functions and projection-to-state safety. In *Learning for Dynamics and Control*, pages 1041–1053. PMLR, 2021.

- Jared Di Carlo, Patrick M Wensing, Benjamin Katz, Gerardo Bledt, and Sangbae Kim. Dynamic locomotion in the mit cheetah 3 through convex model-predictive control. In *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 1–9. IEEE, 2018.
- Ruben Grandia, Farbod Farshidian, René Ranftl, and Marco Hutter. Feedback mpc for torque-controlled legged robots. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4730–4737. IEEE, 2019.
- Nicolas Heess, Dhruva TB, Srinivasan Sriram, Jay Lemmon, Josh Merel, Greg Wayne, Yuval Tassa, Tom Erez, Ziyu Wang, SM Eslami, et al. Emergence of locomotion behaviours in rich environments. *arXiv preprint arXiv:1707.02286*, 2017.
- Ayonga Hereid and Aaron D. Ames. Frost: Fast robot optimization and simulation toolkit. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, Vancouver, BC, Canada, September 2017. IEEE/RSJ.
- Shao-Chen Hsu, Xiangru Xu, and Aaron D Ames. Control barrier function based quadratic programs with application to bipedal robotic walking. In *2015 American Control Conference (ACC)*, pages 4542–4548. IEEE, 2015.
- Jemin Hwangbo, Joonho Lee, and Marco Hutter. Per-contact iteration method for solving contact dynamics. *IEEE Robotics and Automation Letters*, 3(2):895–902, 2018. doi: 10.1109/LRA.2018.2792536.
- Alberto Isidori. Elementary Theory of Nonlinear Feedback for Multi-Input Multi-Output Systems. In Alberto Isidori, editor, *Nonlinear Control Systems*, Communications and Control Engineering, pages 219–291. Springer, London, 1995. ISBN 978-1-84628-615-5. doi: 10.1007/978-1-84628-615-5_5. URL https://doi.org/10.1007/978-1-84628-615-5_5.
- Scott Kuindersma, Robin Deits, Maurice Fallon, Andrés Valenzuela, Hongkai Dai, Frank Permenter, Twan Koolen, Pat Marion, and Russ Tedrake. Optimization-based locomotion planning, estimation, and control design for the atlas humanoid robot. *Autonomous robots*, 40(3):429–455, 2016.
- Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science Robotics*, 5(47):eabc5986, October 2020. doi: 10.1126/scirobotics.abc5986. URL <https://www.science.org/doi/10.1126/scirobotics.abc5986>. Publisher: American Association for the Advancement of Science.
- Yang Liu, Jeremy Bernstein, Markus Meister, and Yisong Yue. Learning by turning: Neural architecture aware optimisation. In *International Conference on Machine Learning*, pages 6748–6758. PMLR, 2021.
- Quan Nguyen and Koushil Sreenath. Safety-critical control for dynamical bipedal walking with precise footstep placement**this work is partially supported through funding from the google faculty award and nsf grant iis-1464337. *IFAC-PapersOnLine*, 48(27):147–154, 2015. ISSN 2405-8963. doi: <https://doi.org/10.1016/j.ifacol.2015.11.167>. URL <https://www.sciencedirect.com>.

com/science/article/pii/S2405896315024258. Analysis and Design of Hybrid Systems ADHS.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc., 2019.

Jerry E Pratt and Sergey V Drakunov. Derivation and application of a conserved orbital energy for the inverted pendulum bipedal walking model. In *Proceedings 2007 IEEE International Conference on Robotics and Automation*, pages 4653–4660. IEEE, 2007.

Marc H. Raibert and Ernest R. Tello. Legged robots that balance. *IEEE Expert*, 1(4):89–89, 1986. doi: 10.1109/MEX.1986.4307016.

Ivan Dario Jimenez Rodriguez, Aaron D. Ames, and Yisong Yue. Lyanet: A lyapunov framework for training neural odes, 2022. URL <https://arxiv.org/abs/2202.02526>.

Nicola Scianca, Daniele De Simone, Leonardo Lanari, and Giuseppe Oriolo. Mpc for humanoid gait generation: Stability and feasibility. *IEEE Transactions on Robotics*, 36(4):1171–1188, 2020.

Jonah Siekmann, Yesh Godse, Alan Fern, and Jonathan W. Hurst. Sim-to-real learning of all common bipedal gaits via periodic reward composition. *CoRR*, abs/2011.01387, 2020. URL <https://arxiv.org/abs/2011.01387>.

Russ Tedrake, Scott Kuindersma, Robin Deits, and Kanako Miura. A closed-form solution for real-time zmp gait generation and feedback stabilization. In *2015 IEEE-RAS 15th International Conference on Humanoid Robots (Humanoids)*, pages 936–940. IEEE, 2015.

Eric R. Westervelt, Jessy W. Grizzle, Christine Chevallereau, Jun Ho Choi, and Benjamin Morris. *Feedback Control of Dynamic Bipedal Robot Locomotion*. CRC Press, 1 edition, October 2018. ISBN 978-1-315-21942-4. doi: 10.1201/9781420053739. URL <https://www.taylorfrancis.com/books/9781420053739>.

Xiaobin Xiong and Aaron Ames. 3d underactuated bipedal walking via h-lip based gait synthesis and stepping stabilization. *arXiv preprint arXiv:2101.09588*, 2021.

Xiaobin Xiong and Aaron D Ames. Orbit characterization, stabilization and composition on 3d underactuated bipedal walking via hybrid passive linear inverted pendulum model. In *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pages 4644–4651. IEEE, 2019.