2020 59th IEEE Conference on Decision and Control (CDC)
Jeju Island, Republic of Korea, December 14-18, 2020

Learning Min-norm Stabilizing Control Laws for Systems with Unknown Dynamics

Tyler Westenbroek*[1], Fernando Castañeda*[2], Ayush Agrawal[2], S. Shankar Sastry[1], Koushil Sreenath[2]

Abstract— **This paper introduces a framework for learning a minimum-norm stabilizing controller for a system with unknown dynamics using model-free policy optimization methods. The approach begins by first designing a Control Lyapunov Function (CLF) for a (possibly inaccurate) dynamics model for the system, along with a function which specifies a minimum acceptable rate of energy dissipation for the CLF at different points in the state-space. Treating the energy dissipation condition as a constraint on the desired closed-loop behavior of the real-world system, we use penalty methods to formulate an unconstrained optimization problem over the parameters of a learned controller, which can be solved using model-free policy optimization algorithms using data collected from the plant. We discuss when the optimization learns a stabilizing controller for the real world system and derive conditions on the structure of the learned controller which ensure that the optimization is strongly convex, meaning the globally optimal solution can be found reliably. We validate the approach in simulation, first for a double pendulum, and then generalize the framework to learn stable walking controllers for underactuated bipedal robots using the Hybrid Zero Dynamics framework. By encoding a large amount of structure into the learning problem, we are able to learn stabilizing controllers for both systems with only minutes or even seconds of training data.**

I. Introduction

Recently, the literature has displayed a renewed interest in data-driven methods for controller design [1]–[4]. Much of this excitement has been driven by recent advances in the model-free reinforcement learning literature [5], [6]. Despite their generality, model-free policy optimization methods are known to suffer from poor sample complexity, as they generally are unable to take advantage of known structure in the control system. This paper bridges the gap between model-based and model-free design paradigms by embedding Lyapunov-based design techniques into a model-free reinforcement learning problem. By encoding basic information about the structure of the system into the learning problem through a Control Lyapunoc Function (CLF), our approach is able to learn optimal stabilizing controllers for highly uncertain systems with as little as seconds or a few minutes of data.

Specifically, the paper proposes a framework for learning a min-norm stabilizing control law for an unknown system using model-free policy optimization techniques. Our approach begins by first designing a CLF for a nominal dynamics model of the system alongside a function which specifies the desired rate of convergence for the closed-loop system. To impose this desired behavior on the real world control system, we then formulate a continuous-time optimization problem over the parameters of a learned controller which treats the energy dissipation condition as a constraint. The cost function for the optimization encourages choices of parameters which minimize control effort, but uses a penalty term to ensure that the dissipation constraint is satisfied, if possible, when the penalty term is chosen to be large enough. The terms in the optimization depend on the dynamics of the unknown system, but discrete-time approximations to the problem can be solved using policy-optimization algorithms and data collected from the plant. In general, the problem may be non-convex, but when the learned controller is linear in its parameters the problem becomes (strongly) convex, meaning the globally optimal solutions for the problem can be found using standard policy gradient [7] or random search techniques [8].

To demonstrate the utility of the proposed framework, we apply the method in simulation to a double pendulum and a high-dimensional model of a bipedal robot. For the double pendulum example, the learned controller is comprised of a linear combination of radial basis functions so that the convexity result discussed above applies, and we demonstrate empirically that the learned controller is able to closely match the true min-norm controller performance. The walking example demonstrates how to extend our results in the body of the paper to encompass the Hybrid Zero Dynamics framework as in [9]. For this high-dimensional system, a feed-forward neural network is used for the learned controller. While we cannot guarantee that the optimal set of parameters is found, the learned controller still produces a stable walking motion in the face of high model uncertainty.

A. Related Work

CLF-based controllers [10], [11] have been proved to be effective for a wide variety of complex robotic tasks, such as bipedal walking [12], [9], manipulation [13] and multi-agent coordination [14]. In [12] and [13] quadratic programs (CLF-QP), which integrate the CLF condition as a constraint, are used to get optimal min-norm stabilizing controllers. The

* Indicates equal contribution.

[1]Department of Electrical Engineering and Computer Sciences, University of California at Berkeley, USA.

[2]Department of Mechanical Engineering, University of California at Berkeley, USA.

The work of Fernando Castañeda received the support of a fellowship (code LCF/BQ/AA17/11610009) from "la Caixa" Foundation (ID 100010434). This work was partially supported through National Science Foundation Grants CMMI-1931853 and CMMI-1944722 and by HICON-LEARN (design of HIgh CONfidence LEARNing-enabled systems), Defense Advanced Research Projects Agency award number FA8750-18-C-0101, and Provable High Confidence Human Robot Interactions, Office of Naval Research award number N00014-19-1-2066.

CLF-QP is solved online and additional constraints, such as input saturation, can be added.

However, the dynamics of many real-world systems have nonlinearities that might be difficult to model correctly and/or physical parameters which could be difficult to identify. Input-to-state stability has been used to tackle this problem in [15], [16]. Also, adaptive [17] and robust [18], [19] versions of CLF-based controllers have been developed in recent years. However, these approaches sometimes fail to account for the correct amount of uncertainty due to the typical assumptions they make on the uncertainties' structures and bounds.

Our work most closely aligns with recent research that use data-driven approaches to tackle the issue of model uncertainty in nonlinear controllers. Our works builds on [20], [21], where reinforcement learning is used to account for uncertainty when performing feedback linearization of nonlinear systems. In contrast to recently proposed approaches [1], [22] which focus on learning the uncertain terms in a CLF-QP in order to indirectly improve the optimization-based controller, the framework proposed in this paper directly learns the optimal stabilizing controller. By directly learning the desired controller, our approach removes the need for solving a real-time optimization problem involving a potentially complex learned component, which may take a non-trivial amount of time to process during real-time applications. On hardware, CLF-based controllers frequently need to be updated at frequencies exceeding 1000 Hertz to maintain the stability of the system [9], placing strict timing requirements on the rate at which the CLF-based controller must be updated. Thus, we hypothesize that the direct approach may have meaningful advantages in applications, and future work will seek to validate this claim.

B. Organization

The rest of the paper is organized as follows. Section II revisits Control Lyapunov Functions. Section III presents the proposed learning problem, develops our theoretical guarantees, and then demonstrates how the discrete-time approximations to the problem can be solved using reinforcment learning in an approach similar to [20], [21]. In Section IV the proposed method is used to stabilize a double pendulum and the walking gait of an underactuated nonlinear bipedal robot. Finally, Section V provides concluding remarks.

C. Notation and Terminology

We let $\mathbb{R}_{\geq 0} = \{x \in \mathbb{R} \colon x \geq 0\}$ denote the closed right half plane. We say that a function $V \colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is *positive definite* if $V(0) = 0$ and $V(x) > 0$ if $x \neq 0$. We further say that V is *radially unbounded* if $V(x) \to \infty$ as $\|x\| \to \infty$. If it exists, the gradient of a function $V \colon \mathbb{R}^n \to \mathbb{R}$ at the point $x \in \mathbb{R}^n$ is denoted by the row vector $\nabla V(x) \in \mathbb{R}^{1 \times n}$.

II. Control Lyapunov Functions

Throughout the paper we will consider nonlinear control-affine systems of the form

$$\dot{x} = f(x) + g(x)u, \tag{1}$$

where $x \in \mathbb{R}^n$ is the state and $u \in \mathbb{R}^m$ the input. The mappings $f \colon \mathbb{R}^n \to \mathbb{R}^n$ and $g \colon \mathbb{R}^n \to \mathbb{R}^{n \times m}$ are assumed to be locally Lipschtiz continuous with $f(0) = 0$.

Definition 1: We say that a continuously differentiable, positive definite, radially unbounded function $V \colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is a *Control Lyapunov Function* (CLF) for (1) with positive definite energy dissipation rate $\sigma \colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ if for each $x \in \mathbb{R}^n \setminus \{0\}$

$$\inf_{u \in \mathbb{R}^m} \nabla V(x) \cdot [f(x) + g(x)u] \leq -\sigma(x). \tag{2}$$

It is well-known that if the above conditions are satisfied then the system is asymptotically controllable in the sense that the state can be driven to the origin asymptotically for every initial condition [11]. For many physical systems it is desirable to find a locally Lipschitz continuous feedback rule $u \colon \mathbb{R}^n \to \mathbb{R}^m$ so that for each $x \in \mathbb{R}^n \setminus \{0\}$

$$\nabla V(x) \cdot [f(x) + g(x)u(x)] \leq -\sigma(x) \tag{3}$$

and the closed loop system is asymptotically stable. It should be noted that not all systems which satisfy (2) admit such a controller [23], but a number of important systems such as the ones considered in this document are continuously stabilizable. One popular choice of control law which satisfies the dissipation constraint (3) is the min-norm control law $u^* \colon \mathbb{R}^n \to \mathbb{R}^m$ which is defined point-wise by:

$$\begin{aligned} u^*(x) = \underset{u \in \mathbb{R}^m}{\arg\min} \; & \|u\|_2^2 \\ \text{s.t. } & \nabla V(x) \cdot [f(x) + g(x)u] \leq -\sigma(x) \end{aligned} \tag{4}$$

At every point, this controller selects the smallest input which ensures that the CLF decays at the desired rate. If V is a CLF for the system, a sufficient condition for u^* to be locally Lipschitz continuous is that f, g and the gradient of V are each locally Lipschitz continuous [24]. Moreover, when there are no constraints on the input, one can derive a closed-form expression for the controller (see e.g. [9]). However, one advantage of formulating the min-norm controller as a point-wise optimization as in (4) is that the optimization can easily incorporate bounds on the allowable control efforts for the system by restricting $u \in U \subset \mathbb{R}^m$. This is important in many applications where the actuators of the system have physical limitations. Our theoretical results rely on the input being unconstrained, however, in practice such constraints can be added by restricting the range-space of the learned controller.

III. Learning Min-norm Stabilizing Controllers

A. Learning a Min-norm Stabilizing Controller for a System with Unknown Dynamics

Despite the wide-spread utility of the CLF-based controllers introduced in the previous section, the primary drawback of these methods is that they require an accurate dynamics model to implement. Our present objective is to

learn a min-norm stabilizing controller for the plant with unknown dynamics

$$\dot{x} = f_p(x) + g_p(x)u, \tag{5}$$

while ensuring that the learned controller adheres to the dissipation constraint imposed by some candidate CLF $V\colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ and associated decay rate $\sigma\colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$.

We will focus on learning the min-norm controller for the plant on a compact subset of the state-space. Specifically, we will focus on learning the min-norm stabilizing controller for the system on the set

$$W^c \coloneqq \{x \in \mathbb{R}^n \colon V(x) \leq c\}, \tag{6}$$

where $c > 0$ is a design parameter.

We will make the following technical assumptions throughout the paper unless otherwise specified:

Assumption 1: The components f_p, g_p, σ and ∇V are each locally Lipschitz continuous.

Assumption 2: There exists a locally Lipschitz continuous control law $\tilde{u}_p\colon \mathbb{R}^n \to \mathbb{R}^m$ such that for each $x \in W^c$

$$\nabla V(x)[f_p(x) + g_p(x)\tilde{u}_p(x)] \leq -\sigma(x). \tag{7}$$

Remark 1: Assumption 2 ensures that V is a true CLF for the system with associated dissipation rate σ. While our approach does not explicity require a nominal dynamics model for the plant, in practice, our candidate CLF for the plant is constructed using a nominal dynamics model

$$\dot{x} = f_m(x) + g_m(x)u, \tag{8}$$

which incorporates any information we have about the plant, but may be inaccurate due to nonlinearities which are difficult to model or dynamics parameters which are challenging to identify. However, despite model mismatch between (5) and (8), we can often design a CLF for the model and reasonably expect it to also be a CLF for the plant. For example, our two numerical examples systematically construct CLF's for the unknown system using feedback-linearizing coordinates. In these examples Assumption 2 is tantamount to knowing the relative degree of the system, a rather mild structural assumption.

Since we do not know the terms in (5), we now propose a method to learn a stabilizing CLF-based controller for the system using data collected from the plant. Under the preceding assumptions, and recalling our discussion from Section II, we know that there is a well-defined control law $u_p^*\colon W^c \to \mathbb{R}_{\geq 0}$ which asymptotically stabilizes the plant on W^c and is given point-wise by

$$\begin{aligned} u_p^*(x) = &\underset{u \in \mathbb{R}^m}{\arg\min} \|u\|_2^2 \\ &\text{s.t. } \nabla V(x)[f_p(x) + g_p(x)u] \leq -\sigma(x). \end{aligned}$$

We will denote our learned approximation for u_p^* by $\hat{u}\colon \mathbb{R}^n \times \Theta \to \mathbb{R}^m$. For each choice of parameter $\theta \in \Theta \subset \mathbb{R}^K$ the control law $\hat{u}(\cdot, \theta)\colon \mathbb{R}^n \to \mathbb{R}^m$ defines the learned control law supplied to the plant, with $\Theta \subset \mathbb{R}^K$ a convex set of allowable learned parameters. It is assumed that $\hat{u}$ is locally Lipschitz continuous in its first argument and continuously differentiable in its second argument. Common function approximators such as feed-forward neural networks, radial basis functions or bases of polynomials can be used to construct the learned controller.

Remark 2: In general, the learned controller can incorporate information from a nominal dynamics model by giving it the structure

$$\hat{u}(x, \theta) = u_m(x) + \delta u(x, \theta), \tag{9}$$

where u_m is a nominal model-based controller and $\delta u\colon \mathbb{R}^n \times \mathbb{R}^K \to \mathbb{R}^m$ is the learned component.

Next, in order to find parameters for the learned controller which satisfy the dissipation constraint (7), we will solve optimizations over the parameters of the learned controller of the form

$$(\mathbf{P}_\lambda) : \min_{\theta \in \Theta} L_\lambda(\theta), \tag{10}$$

where for each $\lambda \in \mathbb{R}_{\geq 0}$ we define the loss function

$$L_\lambda(\theta) = E_{x \sim X}\left[\|u(x, \theta)\|_2^2 + \lambda H(\Delta(x, \theta))\right], \tag{11}$$

where X is the uniform probability distribution over W^c, the mapping $\Delta\colon \mathbb{R}^n \times \Theta \to \mathbb{R}$ is defined by

$$\Delta(x, \theta) = \nabla V(x)[f_p(x) + g_p(x)\hat{u}(x, \theta)] + \sigma(x) \tag{12}$$

and finally $H\colon \mathbb{R} \to \mathbb{R}_{\geq 0}$ is defined for each $y \in \mathbb{R}$ by

$$H(y) = \begin{cases} y & \text{if } y \geq 0, \\ 0 & \text{if } y < 0. \end{cases} \tag{13}$$

The first term in the loss L_λ encourages small control efforts while the second term penalizes violations of the CLF dissipation constraint, with $\lambda \in \mathbb{R}_{\geq 0}$ used to control the magnitude of the penalty. While we do not know $\Delta(x, \theta)$ *a priori*, we can measure this quantity by applying the control $\hat{u}(x, \theta)$ to the plant at the point x and measuring the resulting time derivative of V. Then, equation (12) can be used to compute the desired quantity. Thus, any stochastic optimization algorithm can be used to solve $\mathbf{P}_\lambda$ by running experiments to evaluate the terms in L_λ. We will discuss this in further detail when we present practical approaches for solving $\mathbf{P}_\lambda$ below.

Remark 3: The uniformity of the distribution X ensures that all points in W^c are considered when optimizing over the parameters of $\hat{u}$. This requirement is seen to be analogous to the persistency of excitation conditions which are common in the adaptive control literature [25]. In the proofs of the following theoretical results this condition ensures that each component of the learned controller is activated sufficiently during the learning process.

B. Theoretical Results

We now study how the solution set of $\mathbf{P}_\lambda$ changes as the penalty parameter λ is increased and derive conditions under which the problem is convex, meaning that it can be solved reliably to global optimality using iterative gradient-based

optimization algorithms. To simplify the statement of our results, for each $\lambda \in \mathbb{R}_{\geq 0}$ we define

$$S_\lambda \colon = \left\{ \theta \in \Theta \colon \theta \in \arg\min_{\theta \in \Theta} L_\lambda(\theta) \right\} \tag{14}$$

to capture the set of global minimizers for $\mathbf{P}_\lambda$. We also define

$$\Xi \colon = \{\theta \in \Theta \colon \Delta(x, \theta) \leq 0, \ \forall x \in W^c\} \subset \Theta \tag{15}$$

to be the set of parameters for which the corresponding learned controller satisfies the desired CLF dissipation constraint at every point in W^c. Next, we present our theoretical results in Lemma 1 and Theorems 1 and 2, whose proofs can be found in the Appendix.

First, we compare the sets Ξ and S_λ as the penalty term λ is increased:

Lemma 1: Assume that Ξ is non-empty so that there exists at least one choice of learned parameters which satisfy the desired CLF constraint. Then there exists $\bar{\lambda} \in \mathbb{R}_{\geq 0}$ such that for each $\lambda > \bar{\lambda}$ all global optimizers of $\mathbf{P}_\lambda$ also satisfy the dissipation constraint, namely, $S_\lambda \subset \Xi$.

In other words, if the penalty parameter $\lambda \in \mathbb{R}_{\geq 0}$ is chosen to be large enough then $\mathbf{P}_\lambda$ recovers the set of learned parameters which stabilize the plant and satisfy the CLF constraint. Note that if $\theta^* \in \Xi$ is one such choice of parameters then it must be the case that $\mathbb{E}_{x \sim X}[\lambda H(\Delta(x, \theta^*))] = 0$. Thus, when Ξ is non-empty and λ is chosen to be large enough the minimizers of $\mathbf{P}_\lambda$ are selected by the set of parameters which minimize the term $\mathbb{E}_{x \sim X}\left[\|u(x, \theta)\|_2^2 \right]$, which is the average control effort exerted over the state-space by the corresponding learned controller. By definition, the min-norm stabilizing controller u_p^* minimizes the control effort needed to satisfy the CLF dissipation constraint at every point in the state-space. Thus, if λ is large enough and u_p^* is in the space of learned controllers spanned by $\hat{u}$, it must be recovered by the optimization:

Theorem 1: Assume that there exists $\bar{\theta} \in \Theta$ such that $\hat{u}(x, \bar{\theta}) = u_p^*(x)$ for each $x \in W^c$. Then there exists $\bar{\lambda} \in \mathbb{R}_{\geq 0}$ such that for each $\lambda > \bar{\lambda}$ and $\theta^* \in S_\lambda$ we have $\hat{u}(x, \theta^*) = u_p^*(x)$ for each $x \in W^c$.

However, the family of optimization problems we have formulated over the parameters of the learned controller will generally be non-convex, meaning that we cannot efficiently find their globally optimal solutions. Thus, we seek conditions under which $\mathbf{P}_\lambda$ becomes convex so that we can reliably find its global minimizers using iterative methods. Towards this end we will now assume that

$$\hat{u}(x, \theta) = \sum_{k=1}^{K} \theta_k u_k(x), \tag{16}$$

where $\{u_k\}$ is a set of locally Lipschitz continuous mappings from $\mathbb{R}^n$ to $\mathbb{R}^m$ and θ_k is the k-th entry of the learned parameter[1]. Linearity assumptions of this sort are common in convergence proofs found in both the adaptive control and reinforcement learning literature. In the statement of the following result we informally view each basis element u_k as a subset of $C(W^c, \mathbb{R}^m)$, the vector space of continuous functions from W^c to $\mathbb{R}^m$.

Lemma 2: Assume that $\hat{u}$ is of the form (16) and that the set $\{u_k\}_{k=1}^K$ is linearly independent on $C(W^c, \mathbb{R}^m)$. Then for each $\lambda \in \mathbb{R}_{\geq 0}$ the loss L_λ is strongly convex.

This leads to the main theoretical result of this paper, which follows from an immediate application of Theorem 1 and Lemma 2.

Theorem 2: Assume that u is of the form (16) and that the set $\{u_k\}_{k=1}^K$ is linearly independent on $C(W^c, \mathbb{R}^m)$. Further assume that $\Theta \subset \mathbb{R}^K$ is convex and that there exists $\theta^* \in \Theta$ such that $\hat{u}(x, \theta^*) = u_p^*(x)$ for each $x \in W^c$. Then there exists $\bar{\lambda} \in \mathbb{R}_\lambda \geq 0$ such that for each $\lambda > \bar{\lambda}$ the problem $\mathbf{P}_\lambda$ is a strongly convex optimization problem with θ^* its unique global minimizer.

In practice, since we do not have a parametric model for the system, it is unlikely that there exists a set of learned parameters which exactly reconstructs the true min-norm controlller for the plant. However, many model-free learning schemes [26] make use of function approximation schemes which can recover a continuous function up to a desired level of accuracy, if enough terms are included in the basis of features. It is a matter for future work to show that we can leverage these results to learn $u^*(x)$ up to a pre-specified degree of accuracy. However, in practice the number of elements required in such an expansion can quickly become prohibitively large as the dimension of the state grows. Thus, for high dimensional systems, such as the bipedal robots we consider below, practical implementations may require the use of more compactly represented function approximation schemes, such as multi-layer feed-forward neural networks, which can also approximate continuous functions to a desired degree of accuracy (Universal Approximation Theorem [27], [28]), but lead to non-convexities in our optimization problem (16).

C. Solving Discrete-time Approximations with Reinforcement Learning

Many real-world systems have digital sensors and actuators which can only be updated at some maximum frequency, meaning we can only obtain finite difference approximations of $\dot{V}$ when different control signals are applied to the plant. Thus, in this section we introduce discrete-time approximations to $\mathbf{P}_\lambda$ and discuss how they can be solved with standard reinforcement learning algorithms [29], [30]. Our description of this process will be brief, since the approach is similar to the one described in [20].

For the reinforcement learning problem we will assume that the control supplied to the plant can only be updated at a fixed minimum sampling period $\Delta t > 0$. We will let $t_k = k \times \Delta t$ for each $k \in \mathbb{N}$ denote the set of sampling intervals. When the control $\hat{u}(x, \theta) \in \mathbb{R}^m$ is applied over the interval $[t_k, t_{k+1}]$ a Taylor expansion can be used to show

[1] Alternatively, one could also assume that the learned controller is of the form $\hat{u} = u_m(x) + \sum_{k=1}^K \theta_k u_k(x)$ if the system designer wishes to augment a known model-based controller as in (9). The statement and proof of Theorem 2 go through with minor modifications in this case.

that

$$\Delta(x,\theta) = \underbrace{\frac{V(x(t_{k+1})) - V(x(t_k))}{\Delta t} + \sigma(x(t_k))}_{:=\tilde{\Delta}(x,\theta)} + O(\Delta t^2). \tag{17}$$

Thus for small Δt we use the loss

$$\tilde{l}_\lambda(x,\theta) = \|u(x,\theta)\|_2^2 + \lambda H(\tilde{\Delta}(x,\theta)). \tag{18}$$

We use this approximate pointwise loss to define the following reinforcement learning problem, which serves as an approximation to $\mathbf{P}_\lambda$:

$$\begin{aligned} \tilde{\mathbf{P}}_\lambda\colon \min_{\theta\in\Theta}\ & E_{x_0\sim X}\left[\sum_{k=0}^{N} \tilde{l}(x_k,\theta)\right] \\ \text{s.t. } x_{k+1} &= x_k + \int_{t_k}^{t_{k+1}} [f(x(t)) + g(x(t))u_k]dt \\ u_k &= \hat{u}(x_k,\theta). \end{aligned} \tag{19}$$

Here, the curve $x\colon \mathbb{R} \to \mathbb{R}^n$ is the trajectory of the plant starting from initial condition $x(0) = x_0$, and $N \in \mathbb{N}$ is the number of time steps in each rollout. Probing noise can be added to the input to encourage exploration, e.g, by instead setting $u_k = \hat{u}(x_k,\theta) + w_k$, where $w_k \sim \mathcal{N}(0,\sigma_w^2 I)$ is zero mean random noise. Note that in the special case that $N = 1$ the cost incurred when solving $\tilde{\mathbf{P}}_\lambda$ approaches the cost incurred when solving $\mathbf{P}_\lambda$ as $\Delta t \to 0$. In future work we plan to more formally study the relationship between these two problems. The policy optimization problem (19) is in a standard form for reinforcement learning problems [7], and in the following section we demonstrate how these discrete-time approximations can be used to successfully learn stabilizing controllers for unknown systems.

IV. Examples

A. Double Pendulum

We first use our approach to learn a controller which stabilizes the double pendulum depicted in Figure 2 to the upright position, using the input-output linearization-based CLF design approach introduced in [9] to design the candidate CLF for the learning problem. The system has two generalized coordinates $q = (q_1, q_2)$ which represent the angles that each of the arms make with the vertical, with a motor attached at each joint so the system is fully actuated by torques $\tau = (\tau_1, \tau_2)$. The precise dynamics of the model can be found in [31], with states $x = (q, \dot{q})$ and input $u = \tau$. Below we use the configuration variables as outputs when applying the linearization-based CLF design [9].

As depicted in Figure 2a) the system is parameterized by the masses of the two arms m_1, m_2 as well as their lengths, $l_1, l_2 \in \mathbb{R}$. For the purposes of simulation, we set $m_1 = m_2 = l_1 = l_2 = 1$. To set up the learning problem, we assume that we are given an inaccurate dynamics model with inaccurate estimates $\hat{m}_1, \hat{m}_2, \hat{l}_1, \hat{l}_2$. Specifically, we set $\hat{m}_1 = \hat{m}_2 = \hat{l}_1 = \hat{l}_2 = \frac{1}{2}$ so that each of the parameter estimates are half of their true value.

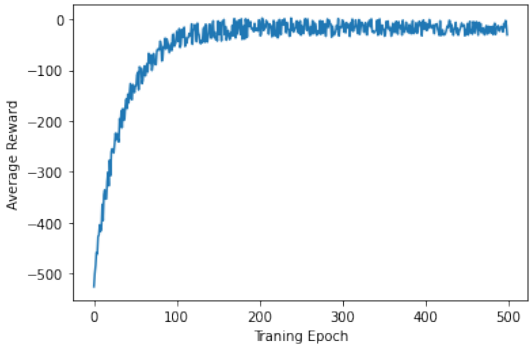

Fig. 1: Learning curve for the double pendulum.

Using the input-output linearization design technique from [9], we design a CLF for the system of the form $V\colon \mathbb{R}^4 \to \mathbb{R}$, $V := x^T P x$, with

$$P = \begin{bmatrix} 1.5I & 0.5I \\ 0.5I & 0.5I \end{bmatrix}, \tag{20}$$

where I is the 2×2 identity matrix and by setting the desired dissipation rate to be $\sigma(x) = x^T x$. This can be shown to be a valid CLF for both the inaccurate dynamics model and the true plant. We focus on learning the min-norm controller for the plant on the set $W^c = \{V(x) \le c\}$ with the design parameter $c = 2$ and construct our learned controller by setting

$$\hat{u}(x,\theta) = u_m(x) + \delta u(x,\theta), \tag{21}$$

where u_m is the min-norm CLF controller computed using the inaccurate dynamic parameters and the learned augmentation δu is comprised of a linear combination of 500 radial basis functions so as to match the assumptions of Lemma 2.

We trained the learned component using a policy-gradient algorithm with action conditioned baselines [7]. Each training epoch consisted of 50 1-step roll-outs and a total of 500 epoch were used. The time-step for the simulator was 0.05 seconds. The performance of the ultimate learned controller is depicted in Figure 2, where we see that the learned controller closely matches the behavior of the true min-norm controller for the system. To further evaluate the performance of the learned controller, we randomly selected 1000 states $\{x_i\}_{i=1}^{1000}$ in W^c and calculated the ratio

$$R = \sum_{i=1}^{1000} \frac{\left\|\hat{u}(x_i,\theta^*) - u_p^*(x_i)\right\|_2}{\left\|u_p^*(x_i)\right\|_2}, \tag{22}$$

where u_p^* is the true min-norm controller for the system and θ^* is the parameter selected by the training process. We calculated $R = 0.044$, indicating that the learned controller was able to closely match the performance of the true min-norm controller for the system. As depicted in Figure 1, the learning converges in about 200 iterations, which corresponds to about eight minutes of data. Our implementation of the learning algorithm for this problem was hand-coded, and we believe the sample efficiency for this problem could match that of the walking example below by improving the implementation.

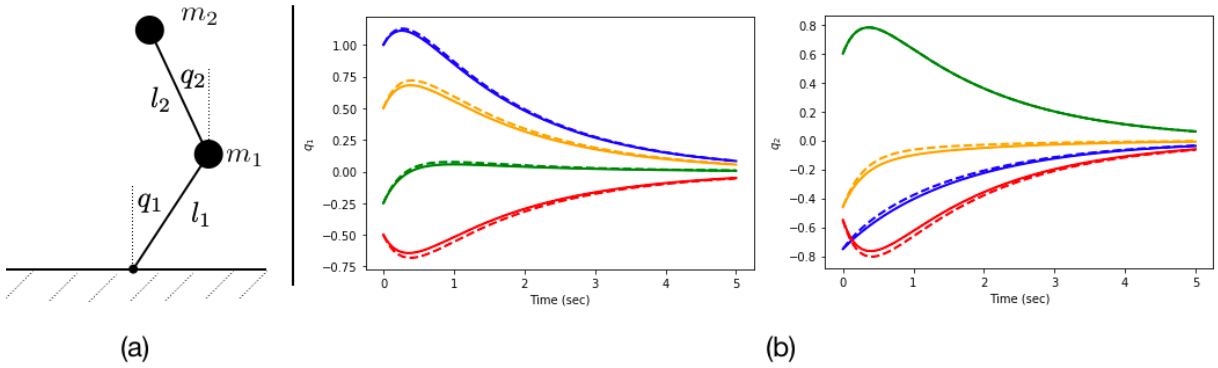

Fig. 2: (a) Depiction of the double pendulum model with the states and physical parameters shown. (b) Trajectories corresponding to different initial conditions for the learned controller and true min-norm controller for the system. Each color represents trajectories starting from a specific initial condition. Solid lines denote the trajectories generated by the true min-norm controller for the system while the dashed lines correspond to the trajectories generated by the learned controller. Observe that the learned controller closely matches the desired closed-loop behavior. Note that the velocities of the trajectories are not depicted, which is why several of the plotted curves intersect.

B. Bipedal Walking

Next, we discuss how to apply our method to the Hybrid Zero Dynamics (HZD) framework using the CLF-based design approach proposed in [9] in order to learn an efficient, stable walking controller for a bipedal robot. We model the robot as a hybrid system with impulse effects as in [9],

$$\Sigma : \begin{cases} \dot{\eta} = f(\eta, z) + g(\eta, z)u, & \\ \dot{z} = h(\eta, z) & \text{when } (\eta, z) \notin \mathcal{S}, \\ \eta^+ = \Delta_X(\eta^-, z^-), & \\ z^+ = \Delta_Z(\eta^-, z^-) & \text{when } (\eta, z) \in \mathcal{S}, \end{cases} \tag{23}$$

where $\eta \in \mathcal{X} \subset \mathbb{R}^{n_a}$ represents the controlled (actuated) states, $z \in \mathcal{Z} \subset \mathbb{R}^{n_u}$ represents the uncontrolled states and $u \in \mathcal{U} \subseteq \mathbb{R}^m$ represents the control inputs. The model assumes alternating phases of single support, where one foot is off the ground (swing foot) and the other (stance foot) is assumed to remain at a fixed point without slipping. The impact between the swing foot and the ground is modelled as a rigid impact and occurs when $(\eta, z) \in \mathcal{S}$, where $\mathcal{S}$ is a smooth switching manifold. Here, $\eta^+ \in \mathcal{X}$ and $z^+ \in \mathcal{Z}$ represent the post-impact states while $\eta^- \in \mathcal{X}$ and $z^- \in \mathcal{Z}$ denote the pre-impact states.

Following the framework in [9], an input-output linearization based CLF is designed for the actuated coordinates during the continuous portion of the evolution of the state. Namely, we design a Lyapunov function $V : \mathbb{R}^{n_u} \to \mathbb{R}$ and dissipation rate $\sigma : \mathbb{R}^{n_u+n_a} \to \mathbb{R}_{\geq 0}$ such that the following condition holds for each $(\eta, z) \in \mathcal{X} \times \mathcal{Z}$:

$$\inf_{u \in U} \nabla V(\eta)[f(\eta, z) + g(\eta, z)u] \leq -\sigma(\eta, z). \tag{24}$$

Thus, the control objective is to drive only the actuated states to zero. As shown in [9], when the coordinates for the actuated and unactuated portions of the system are chosen correctly the condition $\eta \to 0$ corresponds to the robot converging to a periodic walking gate. The CLF and dissipation rate are designed so that the actuated coordinates are driven to zero fast enough to overcome shocks to the system introduced by the switching condition. We refer the readers to [9] for more details on this procedure.

To accommodate this new objective, our goal is to learn a control law $u : \mathcal{X} \times \mathcal{Z} \times \Theta \to \mathbb{R}^m$ such that

$$\underbrace{\nabla V(\eta)[f_p(\eta, z) + g_p(\eta, z)u(\eta, z, \theta)] + \sigma(\eta, z)}_{:=\hat{\Delta}(\eta, z, \theta)} \leq 0 \tag{25}$$

for each $(\eta, z) \in \mathcal{X} \times \mathcal{Z}$ for our choice of learned parameters $\theta \in \Theta$. Here, f_p and g_p are the terms in true dynamics of the plant, which may differ from the nominal dynamics in (23). To modify our approach to this new setting, for each $\lambda \in \mathbb{R}_{\geq 0}$ we now define the loss

$$\hat{L}_\lambda(\theta) = \mathbb{E}_{(\eta,z)\sim X}\left[\|u(\eta, z, \theta)\|_2^2 + \lambda H(\hat{\Delta}(\eta, z, \theta))\right], \tag{26}$$

where X is now the uniform distribution over $\mathcal{X} \times \mathcal{Z}$. Despite the fact that the CLF is defined only over the lower dimensional state η, the theoretical results from section III-B naturally extend to this case. Moreover, the techniques from section III-C can be used to find local minimizers of $\hat{L}_\lambda$.

In particular, the proposed method is validated on a model for RABBIT [32], an under-actuated five-link planar bipedal robot with seven degrees-of-freedom. Model uncertainty is introduced by scaling the mass of each of RABBIT's links by a factor of two, i.e., the real plant's masses are twice the nominal model's masses. Our learned controlled is

$$\hat{u}(\eta, z, \theta) = u_m(\eta, z) + \delta u(\eta, z, \theta), \tag{27}$$

where u_m is the min-norm CLF controller obtained using the nominal model dynamics. The term $\delta u(\eta, z, \theta) \in \mathbb{R}^4$ takes the form of a Multi-Layer Perceptron (MLP) neural network with 2 hidden layers of width 64 each, tanh activation functions and layer normalization. We use the Soft Actor Critic algorithm [30], an off-policy method, for training the learned policy $\delta u(\eta, z, \theta)$. The training is done on episodes consisting of one walking step each. The simulations are conducted on the open-source physics simulator PyBullet [33] using a discrete time-step of one millisecond. As it can be seen in Figure 3, the training converges in about 20,000 time steps, which corresponds to roughly 50 steps of the biped and about 20 seconds of data collection from the system. Altogether, the simulations and training took about

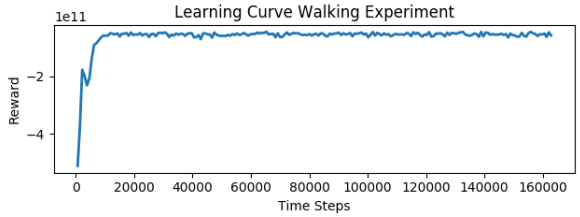

Fig. 3: Learning curve using PyBullet [33] for the walking simulation.

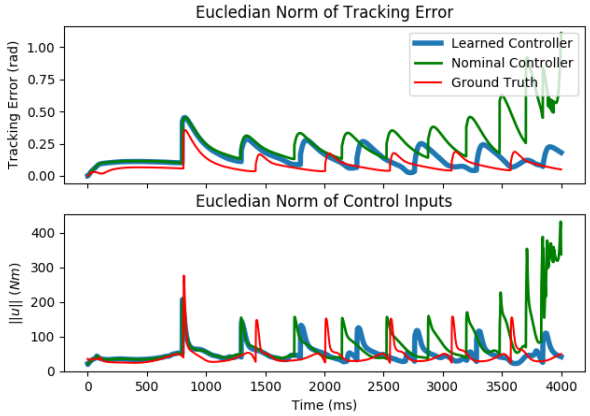

Fig. 4: Tracking error (top) and norm of control inputs (bottom) of the learned min-norm controller (blue), the nominal controller (green) and the actual CLF-based controller of the plant computed using the true robot dynamics (red), each simulated for 4 seconds of walking.

10 minutes of computation using the six cores of an Intel(R) Core(TM) i7-8705G CPU (3.10GHz), without using a GPU.

Figure 4 shows a comparison between the proposed learned controller, the nominal controller u_m and u_p^*, which is the CLF-based controller of the plant computed using the true (unknown) dynamics. This figure shows that while the nominal controller fails after ten walking steps making the robot fall, the learned controller achieves stable walking for an indefinite number of steps and gives good tracking error performance. It is also important to notice that the learned controller achieves this while using similar magnitudes of control inputs as the nominal and the true CLF-based controllers. However, the tracking error performance is not as good as with the actual CLF-based controller of the plant, as expected, and is likely due to the fact that the learner has converged to a local minima. Additionally, we note that the walking speeds for the learned controller and the true min-norm CLF controller for the plant are different. Underactuated robots such as RABBIT may contain multiple periodic orbits on the surface $\{(\eta, z) \in \mathcal{X} \times \mathcal{Z} : \eta = 0\}$. Thus, while both controllers successfully drive the system to this set, the periodic orbits the two controllers converge to are different.

V. Conclusion

There are several important avenues for future work. Demonstration of the proposed method on actual robotic systems in the face of sensor and actuator noise will be required before wider adoption of the technique. Future work will also investigate the inclusion of other local constraints on the dynamics of the closed loop system, such as those imposed by control barrier functions [34]. On the theoretical side, sample complexity guarantees for the methods and a stronger characterization of the trade-off between our approach and model-based methods would more clearly characterize the advantages and disadvantages of the proposed approach.

Appendix

This Appendix contains proofs for several assertions made in the body of the document.

A. Proof of Lemma 1

To prove the desired result, we demonstrate that for each $\theta^* \in \Theta \setminus \Xi$ there exists a finite $\bar{\lambda} \in \mathbb{R}_{\geq 0}$ such that $\theta^* \notin S_\lambda$ for each $\lambda > \bar{\lambda}$. For a fixed $\theta^* \in \Theta \setminus \Xi$, define $M_1^{\theta^*} = E_{x \sim X}[\|\hat{u}(x, \theta^*)\|_2^2]$ and $M_2^{\theta^*} = E_{x \sim X}[H(\Delta(x, \theta^*))]$ so that for each $\lambda > 0$ we have $L_\lambda(\theta^*) = M_1^{\theta^*} + \lambda M_2^{\theta^*}$. Since $\theta^* \notin \Xi$, there must exist $x^* \in W^c$ such that $H(\Delta(x^*, \theta^*)) > 0$. Under our standing assumptions, the map $H(\Delta(\cdot, \theta^*))$ can be seen to be continuous, since the space of continuous functions is closed under addition, multiplication and composition. Putting these two facts together, there must exist a $\delta > 0$ such that for each $x \in B^\delta(x^*) \cap W^c$ we have $H(\Delta(x, \theta^*)) > 0$. This in turn implies that $M_2^{\theta^*} > 0$. Thus, we see that $L_\lambda(\theta^*) \to \infty$ as $\lambda \to \infty$.

Next, letting $\bar{\theta}$ be defined as in the statement of the lemma, for each $\lambda \in \mathbb{R}_{\geq 0}$ we have $L_\lambda(\bar{\theta}) = M_1^{\bar{\theta}}$ where $M_1^{\bar{\theta}} = E_{x \sim X}[\|\hat{u}(x, \bar{\theta})\|_2^2]$ and we note that the term $E_{x \sim X}[H(\Delta(x, \theta^*))]$ contributes nothing to $L_\lambda(\bar{\theta})$ since $\bar{\theta} \in \Xi$. Thus, if we set $\bar{\lambda} = \max\left\{0, \frac{M_1^{\bar{\theta}} - M_1^{\theta^*}}{M_2^{\theta^*}}\right\}$ we see that $L_\lambda(\theta^*) > L_\lambda(\bar{\theta})$ for each $\lambda > \bar{\lambda}$, proving the desired statement for our fixed θ^*.

B. Proof of Theorem 1

Let $\bar{\lambda}$ be defined as in the statement of Lemma 1. Then for each $\lambda > \bar{\lambda}$ we have $S_\lambda \subset \Xi$, where Ξ is defined as in (15). This implies that for each $\theta \in S_\lambda$ we have $L(\theta) = E_{x \sim X}[\|u(x, \theta)\|_2^2]$. Let $\bar{\theta}$ be defined as in the statement of the theorem, and let $\theta \in S_\lambda$ be arbitrary. By the definition of the min-norm control law we have $\|\hat{u}(x, \bar{\theta})\|_2 \leq \|\hat{u}(x, \theta)\|_2$ for each $x \in W^c$, which in turn implies that $L(\bar{\theta}) \leq L(\theta)$. Next, suppose that $u(x^*, \theta) \neq u_p^*(x^*)$ for some $x^* \in W^c$. Again, using the definition of u_p^* we have $\|\hat{u}(x^*, \bar{\theta})\|_2 < \|\hat{u}(x^*, \theta)\|_2$. By the continuity of $\hat{u}(\cdot, \theta)$, we know that there exists $\delta > 0$ such that for each $x \in B^\delta(x^*) \cap W^c$ we have $\|\hat{u}(x, \bar{\theta})\|_2^2 < \|\hat{u}(x, \theta)\|_2^2$. This implies that $L(\bar{\theta}) < L(\theta)$, demonstrating the desired result.

C. Proof of Lemma 2

To prove the claim, we will first consider the two maps $\theta \to E_{x \sim X}\left[\|u(x, \theta)\|_2^2\right]$ and $\theta \to E_{x \sim X}[\lambda H(\Delta(x, \theta))]$ separately. In particular, we will show that the first term is strongly convex in θ while the second term is simply convex. The result of the theorem then follows from the fact that the addition of a strongly convex function and a convex function yields a strongly convex function.

First, we rewrite $\|u(x,\theta)\|_2^2$ as $\theta^T W(x)^T W(x)\theta$ where $W(x) = [u_1(x), u_2(x), \ldots, u_K(x)]^T$ collects the basis of control functions. Note that the positive semi-definite matrix $\bar{W} = E_{x\sim X}\left[W(x)^T W(x)\right]$ is the Grammian for $\{u_k\}_{k=1}^K$ on $C(W^c, \mathbb{R}^m)$, and thus will be full-rank and positive definite iff $\{u_k\}_k^K$ is linearly independent on this space. Based on these facts, we see that $E_{x\sim X}\left[\|u(x,\theta)\|_2^2\right] = \theta^T \bar{W}\theta$ is a strongly convex quadratic function of the parameters.

Next, we turn to the term $E_{x\sim X}[\lambda H(\Delta(x,\theta))]$. We demonstrate that for a fixed $x^* \in W^c$ and each $\lambda \in \mathbb{R}_{\geq 0}$ the mapping $\theta \to \|u(x^*,\theta)\|_2^2 + \lambda H(\Delta(x^*,\theta))$ is strongly convex using basic properties of convex functions [35]. We begin by examining the term $H(\Delta(x,\theta))$. Examining equations (16) and (12) we see that the map $\theta \to \Delta(x^*,\theta)$ is affine in θ for each fixed $x^* \in W^c$. Furthermore, we may rewrite the term $\lambda H(y) = \max\{0, \lambda y\}$. Since the pointwise maximum of two affine functions defines a convex function, we see that $\theta \to \lambda H(\Delta(x^*,\theta))$ is convex, implying that

$$\lambda H(\Delta(x, \alpha\theta_3)) \leq \alpha\lambda H(\Delta(x,\theta_1)) + (1-\alpha)\lambda H(\Delta(x,\theta_2))$$

for each $x \in W^c$, $\theta_1, \theta_2 \in \mathbb{R}^K$ and $\theta_3 = \alpha\theta_1 + (1-\alpha)\theta_2$ for some $\alpha \in [0,1]$. This pointwise fact implies that

$$\begin{aligned} E_{x\sim X}[\lambda H(\Delta(x,\theta_3))] \leq \alpha E_{x\sim X}[\lambda H(\Delta(x,\theta_1))] \\ + (1-\alpha)E_{x\sim X}[\lambda H(\Delta(x,\theta_2))]. \end{aligned}$$

Thus, $\theta \to E_{x\sim X}[\lambda H(\Delta(x,\theta))]$ is convex, as desired.

REFERENCES

[1] A. J. Taylor, V. D. Dorobantu, H. M. Le, Y. Yue, and A. D. Ames, "Episodic learning with control lyapunov functions for uncertain robotic systems," 2019.

[2] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause, "Safe model-based reinforcement learning with stability guarantees," in *Advances in neural information processing systems*, 2017, pp. 908–918.

[3] 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*, 2016, pp. 270–277.

[4] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, "Reachability-based safe learning with gaussian processes," in *IEEE Conference on Decision and Control*, 2014, pp. 1424–1431.

[5] J. Hwangbo, I. Sa, R. Siegwart, and M. Hutter, "Control of a quadrotor with reinforcement learning," *IEEE Robotics and Automation Letters*, vol. 2, no. 4, pp. 2096–2103, Oct 2017.

[6] S. Levine, C. Finn, T. Darrell, and P. Abbeel, "End-to-end training of deep visuomotor policies," *J. Mach. Learn. Res.*, vol. 17, no. 1, p. 1334–1373, Jan. 2016.

[7] R. S. Sutton and A. G. Barto, *Reinforcement learning: An introduction*. MIT press, 2018.

[8] H. Mania, A. Guy, and B. Recht, "Simple random search provides a competitive approach to reinforcement learning," *arXiv preprint arXiv:1803.07055*, 2018.

[9] 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.

[10] Z. Artstein, "Stabilization with relaxed controls," *Nonlinear Analysis: Theory, Methods and Applications*, vol. 7, no. 11, pp. 1163 – 1173, 1983.

[11] E. D. Sontag, "A 'universal' construction of artstein's theorem on nonlinear stabilization," *Systems and Control Letters*, vol. 13, no. 2, pp. 117 – 123, 1989.

[12] 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.

[13] A. Ames and M. Powell, "Towards the unification of locomotion and manipulation through control lyapunov functions and quadratic programs," *Lecture Notes in Control and Information Sciences*, vol. 449, pp. 219–240, 01 2013.

[14] P. Ogren, M. Egerstedt, and X. Hu, "A control lyapunov function approach to multi-agent coordination," in *Proceedings of the 40th IEEE Conference on Decision and Control (Cat. No.01CH37228)*, vol. 2, 2001, pp. 1150–1155 vol.2.

[15] J. P. Hespanha, D. Liberzon, and A. R. Teel, "Lyapunov conditions for input-to-state stability of impulsive systems," *Automatica*, vol. 44, no. 11, pp. 2735 – 2744, 2008.

[16] E. D. Sontag, "On the input-to-state stability property," *European Journal of Control*, vol. 1, no. 1, pp. 24 – 36, 1995.

[17] Q. Nguyen and K. Sreenath, "L1 adaptive control for bipedal robots with control lyapunov function based quadratic programs," in *American Control Conference*, Chicago, IL, July 2015, pp. 862–867.

[18] S. Battilotti, "Robust stabilization of nonlinear systems with pointwise norm-bounded uncertainties: a control lyapunov function approach," *IEEE Transactions on Automatic Control*, vol. 44, no. 1, pp. 3–17, 1999.

[19] Q. Nguyen and K. Sreenath, "Optimal robust control for bipedal robots through control lyapunov function based quadratic programs," in *Robotics: Science and Systems*, Rome, Italy, July 2015.

[20] T. Westenbroek, D. Fridovich-Keil, E. Mazumdar, S. Arora, V. Prabhu, S. S. Sastry, and C. J. Tomlin, "Feedback linearization for unknown systems via reinforcement learning," *arXiv preprint arXiv:1910.13272*, 2019.

[21] F. Castañeda, M. Wulfman, A. Agrawal, T. Westenbroek, C. Tomlin, S. Sastry, and K. Sreenath, "Improving input-output linearizing controllers for bipedal robots via reinforcement learning," in *Learning for Dynamics and Control*, Berkeley, CA, June 2020, pp. 990–999.

[22] J. Choi, F. Castañeda, C. Tomlin, and K. Sreenath, "Reinforcement Learning for Safety-Critical Control under Model Uncertainty, using Control Lyapunov Functions and Control Barrier Functions," in *Robotics: Science and Systems*, Corvalis, OR, July 2020.

[23] S. Sastry, *Nonlinear systems: analysis, stability, and control*. Springer Science & Business Media, 1999, vol. 10.

[24] R. Freeman and P. V. Kokotovic, *Robust nonlinear control design: state-space and Lyapunov techniques*. Springer Science and Business Media, 2008.

[25] S. S. Sastry and A. Isidori, "Adaptive control of linearizable systems," *IEEE Transactions on Automatic Control*, vol. 34, no. 11, pp. 1123–1131, 1989.

[26] F. L. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE circuits and systems magazine*, vol. 9, no. 3, pp. 32–50, 2009.

[27] G. Cybenko, "Approximation by superpositions of a sigmoidal function," *Mathematics of Control, Signals and Systems*, vol. 2, no. 4, pp. 303–314, 1989.

[28] K. Hornik, "Approximation capabilities of multilayer feedforward networks," *Neural Networks*, vol. 4, no. 2, pp. 251 – 257, 1991.

[29] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov, "Proximal policy optimization algorithms," *CoRR*, vol. abs/1707.06347, 2017.

[30] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor," *CoRR*, vol. abs/1801.01290, 2018.

[31] E. Capello, H. Park, B. Tavora, G. Guglieri, and M. Romano, "Modeling and experimental parameter identification of a multicopter via a compound pendulum test rig," in *Workshop on Research, Education and Development of Unmanned Aerial Systems*, 2015, pp. 308–317.

[32] C. Chevallereau, G. Abba, Y. Aoustin, F. Plestan, E. R. Westervelt, C. Canudas-De-Wit, and J. W. Grizzle, "Rabbit: a testbed for advanced control theory," *IEEE Control Systems Magazine*, vol. 23, no. 5, pp. 57–79, 2003.

[33] E. Coumans and Y. Bai, "Pybullet, a python module for physics simulation for games, robotics and machine learning," http://pybullet.org, 2016–2019.

[34] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *European Control Conference*, 2019, pp. 3420–3431.

[35] S. Boyd, S. P. Boyd, and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.