Available online at www.sciencedirect.com

ScienceDirect

IFAC PapersOnline CONFERENCE PAPER ARCHIVE

IFAC PapersOnLine 54-5 (2021) 19–24

Combining Model-Based Design and Model-Free Policy Optimization to Learn Safe, Stabilizing Controllers

Tyler Westenbroek† * **Ayush Agrawal**† ** **Fernando Castañeda**† ** **S Shankar Sastry** * **Koushil Sreenath** **

* *Electrical Engineering and Computer Sciences, UC Berkeley*
** *Mechanical Engineering, UC Berkeley*

Abstract: This paper introduces a framework for learning a safe, stabilizing controller for a system with unknown dynamics using model-free policy optimization algorithms. Using a nominal dynamics model, the user specifies a candidate Control Lyapunov Function (CLF) around the desired operating point, and specifies the desired safe-set using a Control Barrier Function (CBF). Using penalty methods from the optimization literature, we then develop a family of policy optimization problems which attempt to minimize control effort while satisfying the pointwise constraints used to specify the CLF and CBF. We demonstrate that when the penalty terms are scaled correctly, the optimization prioritizes the maintenance of safety over stability, and stability over optimality. We discuss how standard reinforcement learning algorithms can be applied to the problem, and validate the approach through simulation. We then illustrate how the approach can be applied to a class of hybrid models commonly used in the dynamic walking literature, and use it to learn safe, stable walking behavior over a randomly spaced sequence of stepping stones.

Keywords: Adaptation and learning in physical agents, Lyapunov methods, Reinforcement learning control, Control problems under conflict and/or uncertainties, Optimal control theory.

1. INTRODUCTION

Following recent empirical successes from the reinforcement learning (RL) literature (Levine et al. (2016)), there has been a renewed interest in data-driven methods for controller design in the case of model uncertainty (Berkenkamp et al. (2017); Akametalu et al. (2014)). However, despite the flexibility of model-free approaches, these methods are known to suffer from poor sample complexity since they do not take advantage of known structural properties of the control system. Moreover, the literature currently lacks constructive methods for designing learning problems which give the system designer fine-grained control over potentially competing global objectives, such as the rate of convergence to a desired operating point or the avoidance of an unsafe region of the state-space.

Fortunately, modern model-based control theory has developed many tools such as Control Lyapunov Functions (CLFs; Sontag (1989)) and Control Barrier Functions (CBFs; Ames et al. (2017)) which allow the system designer to constrain the pointwise closed-loop behavior of a given control system to ensure desired global properties (stability and safety, respectively) are achieved. When an accurate dynamics model is available, online optimization can be used to satisfy these pointwise constraints while minimizing a cost, such as control effort (Ames et al., 2017). In effect, these approaches reduce the satisfaction of challenging global objectives to simple local decisions from the perspective of controller synthesis.

This paper takes preliminary steps towards extending this design philosophy to the model-free setting by introducing a framework for systematically designing policy optimization problems over a parameterized learned controller which enforces a hierarchy of user-specified constraints on the closed-loop dynamics. To make the framework explicit, we focus on learning safe, stabilizing controllers using CLFs and CBFs and choose to prioritize safety over stability. We focus on the regime where the system designer has access to a dynamics model which may be highly inaccurate but is assumed to at least capture basic structural information about the real world plant. The model is used to construct a candidate CLF and CBF for the plant and a family of policy optimization problems are formulated which use penalty terms to discourage violations of the pointwise constraints imposed by these functions. This allows the system designer to carefully constrain the desired closed-loop behavior for the learned controller while also allowing for additional performance terms, such as minimizing control effort.

Our theoretical results demonstrate how to scale the penalty terms to control violations of the constraints and appropriately prioritize safety over stability and stability

[1] † denotes equal contribution

[2] This work was partially supported through NSF Grants CMMI-1931853 and CMMI-1944722, and HICON-LEARN DARPA award number FA8750-18-C-0101. The work of F. Castañeda received the support of a fellowship from Fundación Rafael del Pino, Spain.

10.1016/j.ifacol.2021.08.468

over performance. We first introduce the approach for classical control systems but then demonstrate how to extend the approach to the hybrid case via an application to a class of hybrid models which are frequently used in the dynamic walking literature (Grizzle et al. (2014)). We discuss how to synthesize numerical approximations to the family of learning problems which can be solved using standard machine learning techniques, including state of the art reinforcement learning algorithms. Simulation experiments are provided for both the continuous and hybrid cases, which demonstrate that our method is able to effectively learn safe, stabilizing controllers in the face of large amounts of dynamics uncertainty. We can reliably solve the policy optimization problems formulated over these systems using only a few minutes or even seconds of simulated data, representing a sharp increase in the sample efficiency usually found in the reinforcement learning literature (Hwangbo et al., 2017; Levine et al., 2016). We conjecture that this is due to the large amount of structure embedded in the learning problem through the incorporation of CLF and CBF constraints, which reduce the search for an optimal safe, stabilizing controller to a set of local criteria at each point in the state space.

Related Work: The unification of Control Barrier Functions and Control Lyapunov Functions to synthesize safe, stabilizing controllers was first proposed in Ames et al. (2017) using online quadratic programming. In the case of model uncertainty, robust formulations have been proposed (Nguyen and Sreenath (2021)). Learning based methods using supervised learning (Taylor et al., 2020) or reinforcement learning (Choi et al., 2020) to learn the uncertain dynamics terms in the quadratic program have also been considered. These can be thought of as indirect learning methods, since they still require solving an optimization problem involving the learned components to calculate the desired controller. The primary downside of each of these approaches is that if the optimization is infeasible at a particular point then the control strategy will generally be undefined, which can be particularly difficult to rule out when learning unknown dynamics.

Building on our previous work (Westenbroek et al., 2020), we introduce a framework for directly learning a safe, stabilizing controller for the system using model-free policy optimization algorithms. 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. At points where it is infeasible to satisfy the desired constraints, our method provides a "best effort" control strategy which satisfies the constraints to the greatest degree possible, bypassing issues of feasibility.

Remark 1. A longer version of this paper can be found at Westenbroek et al. (2021).

2. CONTROL LYAPUNOV FUNCTIONS AND CONTROL BARRIER FUNCTIONS

Throughout most of the paper we will consider 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. We assume that $f\colon \mathbb{R}^n \to \mathbb{R}^n$ and $g\colon \mathbb{R}^n \to \mathbb{R}^{n\times m}$ are continuously differentiable.

Control Lyapunov Functions: Control Lyapunov Functions (CLFs; Sontag (1989)) are commonly used to construct a controller which stabilizes a system to either a desired operating point or a desired subset of the state-space (Ames et al., 2014). Specifically, we say that the continuously differentiable function $V\colon \mathbb{R}^n \to \mathbb{R}_{\geq 0}$ is a *Control Lyapunov Function* if

$$\inf_{u\in\mathbb{R}^m} \nabla V(x)[f(x) + g(x)u] \leq -\sigma(x) \quad \forall x \in \mathbb{R}^n \setminus \{0\}, \tag{2}$$

where $\sigma\colon \mathbb{R}^n \to \mathbb{R}$ specifies a desired pointwise rate of decay. Here, V and σ are both assumed to be positive definite, and V is additionally assumed to be radially unbounded. Under these conditions, V can be viewed as a generalized energy function for (1), and condition (2) ensures that there exists a control which drives the system state asymptotically to the origin.

Control Barrier Functions: Inspired by barrier functions from the optimization literature, the level sets of Control Barrier Functions (CBFs) encode user-specified safety constraints. Many classes of CBFs have been proposed in recent years (Ames et al., 2017; Xu et al., 2015; Nguyen and Sreenath, 2016), but for concreteness throughout the paper we will use the class proposed in Xu et al. (2015). Specifically, we say that the function $h\colon \mathbb{R}^n \to \mathbb{R}$ is a *Control Barrier Function* if

$$\sup_{u\in\mathbb{R}^m} \nabla h(x)[f(x) + g(x)u] \geq -\alpha(h(x)) \quad \forall x \in \mathcal{C}, \tag{3}$$

where $\mathcal{C} = \{x \in \mathbb{R}^n : h(x) \geq 0\}$ is a safe set specified by the 0-super-level set of h, and $\alpha\colon (-b, a) \to \mathbb{R}$, with $a, b > 0$, is locally Lipschitz, strictly increasing, and $\alpha(0) = 0$.

It is natural to then search for a Lipschitz continuous control law which satisfies the pointwise constraints in (2) and (3) simultaneously. One candidate control law is given by solving a pointwise quadratic program (QP):

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

which aims to minimize control effort while satisfying the two pointwise constraints. Unfortunately, even if V and h are an actual CLF and CBF for the system, it may be impossible to satisfy both constraints simultaneously leading to infeasibility issues. A common heuristic is to add slack terms to one or both of the constraints to ensure feasibility of the problem at the cost of some violation of the constraints (Ames et al., 2017).

3. LEARNING SAFE, STABILIZING CONTROLLERS FOR UNCERTAIN SYSTEMS

While control laws similar to (4) have been successfully applied in a number of applications they have several practical limitations. Most importantly, these approaches require that an exact dynamics model is available to ensure that the pointwise constraints in (4) can be satisfied on the real-world system. Secondly, the infeasibility issues mentioned above mean that the controller may be undefined at certain points in the state-space, which can be highly

problematic during real-time operation. This motivates the method detailed below, which uses a candidate CLF and CBF to learn an optimal safe, stabilizing controller for an uncertain system using data collected from the plant. The method prioritizes satisfaction of the CBF constraint over the CLF constraint and removes the need for real-time optimization.

Specifically we will seek to safely stabilize the *plant*

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

whose dynamics are unknown. We will also assume that a nominal *dynamics model* for the plant is available:

$$\dot{x} = f_m(x) + g_m(x)u. \tag{6}$$

We assume that the dynamics model has been used to synthesize a candidate CLF V (and rate σ) and CBF h (and rate α) for the unknown plant. Even though the dynamics of the plant are unknown, it is often reasonable to assume that the model captures enough basic structural information about the plant to guarantee that these functions are also a valid CLF and CBF for the real-world system. For example, in our simulated applications we design the candidate CLF using feedback linearization, which is guaranteed to be a CLF for the true system as long as the relative degree of the plant matches that of the model, a relatively weak assumption.

The learned controller $\hat{u}\colon \mathbb{R}^n \times \Theta \to \mathbb{R}^m$ is of the form

$$\hat{u}(x,\theta) = u_m(x) + \tilde{u}(x,\theta). \tag{7}$$

Here, $u_m\colon \mathbb{R}^n \to \mathbb{R}^m$ is a nominal controller supplied by the system designer which is derived from the nominal dynamics model, and $\tilde{u}\colon \mathbb{R}^n \times \Theta \to \mathbb{R}^m$ is a learned augmentation. The learned parameters $(\theta_1, \dots, \theta_p) \in \Theta \subset \mathbb{R}^p$ are to be trained so as to select the optimal safe, stabilizing controller for the system.

Assumption 1. The learned controller $\hat{u}\colon \mathbb{R}^n \times \theta \to \mathbb{R}^m$ is continuously differentiable in both of its arguments.

Assumption 2. The set of learned parameters Θ is a compact convex set.

Our primary goal is to find a controller which satisfies the following infinite dimensional constraints, when possible:

$$\underbrace{-\nabla h(x)[f_p(x) + g_p(x)\hat{u}(x,\theta)] - \alpha(h(x))}_{\Delta_1(x,\theta)} \leq 0 \quad \forall x \in \mathcal{C}, \tag{8}$$

$$\underbrace{\nabla V(x)[f_p(x) + g_p(x)\hat{u}(x,\theta)] + \sigma(x)}_{\Delta_2(x,\theta)} \leq 0 \quad \forall x \in \mathcal{C}. \tag{9}$$

Here, the set $\mathcal{C} \subset \mathbb{R}^n$ is the safe-set defined by the 0-super-level set of h. In words, we want to train a controller $\hat{u}(\cdot,\theta)\colon \mathbb{R}^n \to \mathbb{R}$ which satisfies the safety and stabilization constraints that the chosen CBF and CLF impose on the real-world system. We make the following assumption:

Assumption 3. The safe set $\mathcal{C}$ is compact.

Since it may not be possible to learn a controller which satisfies both sets of constraints simultaneously, our learning framework must be flexible enough to prioritize the safety objective over the stabilization objective when necessary. While we do not know the terms in $\Delta_1(x,\theta)$ and $\Delta_2(x,\theta)$ since the dynamics of the plant are unknown, these terms can be calculated for different values of $x \in \mathcal{C}$ and $\theta \in \Theta$ if measurements of $\dot{V}$ and $\dot{h}$ are available when collecting data from the plant.

In order to enforce these constraints while minimizing control effort, we will solve optimizations of the form

$$\mathbf{P}^{(\lambda_1,\lambda_2)}\colon \ \min_{\theta\in\Theta} \mathbb{E}_{x\sim X} L^{(\lambda_1,\lambda_2)}(x,\theta),$$

where

$$L^{(\lambda_1,\lambda_2)}(x,\theta) = \|\hat{u}(x,\theta)\|_2^2 + \lambda_1 H(\Delta_1(x,\theta)) + \lambda_2 H(\Delta_2(x,\theta)),$$

the hinge map $H\colon \mathbb{R} \to \mathbb{R}$ is defined by $H(y) = \max\{0, y\}$ for each $y \in \mathbb{R}$, and the probability distribution $X\colon \mathcal{C} \to [0,1]$ is supported on $\mathcal{C}$. Here, X is understood to be the distribution of states visited when collecting samples from the real world plant during the learning process, and $\lambda_1, \lambda_2 \geq 0$ are penalty parameters to be chosen later.

Remark 2. The requirement that X is supported on all of $\mathcal{C}$ is analogous to the persistency of excitation conditions found in the adaptive control literature (Sastry and Bodson, 1989), and ensures that the data is "rich enough" so that the correct controller is learned. Note that under this assumption the penalty terms $\mathbb{E}_{x\sim X}\lambda_1 H(\Delta_1(x,\theta))$ and $\mathbb{E}_{x\sim X}\lambda_2 H(\Delta_2(x,\theta))$ are positive if and only if the safety and stability constraints are violated, respectively, at some point $x \in \mathcal{C}$. Thus this richness requirement guarantees that violations of the pointwise constraints are appropriately penalized by the optimization. The theoretical guarantees we provide below are algorithm agnostic, and seek to characterize the global optimizers of the problem. Future work will seek to bound the performance of specific machine learning algorithms used to solve $\mathbf{P}^{(\lambda_1,\lambda_2)}$, which generally come in the form of probabilistic guarantees.

Theoretical Analysis: We now demonstrate that violations of the safety and stability constraints can be decreased to a pre-specified tolerance by scaling the penalty terms appropriately. For simplicity, we assume there exists at least one set of parameters which satisfies the safety constraint:

Assumption 4. There exists $\theta^* \in \Theta$ such that for each $x \in \mathcal{C}$ we have $\Delta_1(x,\theta^*) \leq 0$.

Next, we build up some additional notation to simplify the statement of our theoretical results. First, define the maps $M_u, M_1, M_2\colon \Theta \to \mathbb{R}_{\geq 0}$ by

$$M_u(\theta) = \mathbb{E}_{x\sim X}\|\hat{u}(x,\theta)\|_2^2,$$
$$M_1(\theta) = \mathbb{E}_{x\sim X} H(\Delta_1(x,\theta)),$$
$$M_2(\theta) = \mathbb{E}_{x\sim X} H(\Delta_2(x,\theta)).$$

For each chosen parameter $\theta \in \Theta$, $M_u(\theta)$ captures total energy exerted by the corresponding controller across the safe set, $M_1(\theta)$ is the extent to which the CBF constraint is violated, and $M_2(\theta)$ is the extent to which the CLF constraint is violated. Next, for each $\varepsilon_1 \geq 0$ define

$$\Theta_{\varepsilon_1} = \{\theta \in \Theta\colon M_1(\theta) \leq \varepsilon_1\},$$

which is the set of parameters for which the total violation of the CBF constraint is less than ε_1. We also define

$$\tilde{M}_2 = \min_{\theta\in\Theta_0} M_2(\theta), \tag{10}$$

which is the smallest extent to which the CLF constraint can be violated, subject to exact satisfaction of the CBF constraint, and is the ideal amount of violation of the CLF constraint that can be returned by our optimization problem. We then define for each $\varepsilon_1, \varepsilon_2 \geq 0$

$$\Theta_{\varepsilon_1,\varepsilon_2} = \{\theta \in \Theta_{\varepsilon_1}\colon M_2(\theta) \leq \tilde{M}_2 + \varepsilon_2\},$$

which is the set of parameters corresponding to learned controllers which violate the CBF and CLF constraints no more than $\varepsilon_1 \geq 0$ and $\varepsilon_2 \geq 0$ more than their ideal values.

We now present our first result, whose proof can be found in the Appendix:

Theorem 1. There exist constants, $C_1, C_2, C_3 \geq 0$ such that if $\lambda_1 \geq \frac{C_1\lambda_2 + C_2}{\varepsilon_1}$ and $\lambda_2 \geq \frac{C_3}{\varepsilon_2}$ then each global optimizer θ^* of $\mathbf{P}^{(\lambda_1,\lambda_2)}$ satisfies $\theta^* \in \Theta_{\varepsilon_1,\varepsilon_2}$.

The result indicates that if we choose $\lambda_2 \gg 0$ and $\lambda_1 \gg \lambda_2$ our optimization correctly enforces safety over stability, satisfying the two constraints to the desired tolerances. Within the set of desired controllers specified by $\Theta_{\varepsilon_1,\varepsilon_2}$, the optimization is then left to reduce the amount of control effort required to achieve these objectives. However, driving both tolerances to zero requires taking $\lambda_1, \lambda_2 \to \infty$.

One practical approach for ensuring exact satisfaction of the safety constraint for a finite value of the multipliers is to add a small amount of extra conservativeness to the pointwise CBF constraint. Specifically, letting $\Delta_1^\delta(\theta, x) = \Delta_1(x, \theta) + \delta$ for some small parameter $\delta > 0$, one can replace $\Delta_1(x, \theta)$ with $\Delta_1^\delta(x, \theta)$ in the loss $L^{(\lambda_1,\lambda_2)}(x, \theta)$. Due to the continuity of the problem data, driving $\mathbb{E}_{x\sim X} H(\Delta_1^\delta(\theta, x))$ to be sufficiently small (which can be done with finite values of λ_1) will ensure exact satisfaction of the original CBF constraint. A forthcoming article will address this point in greater detail.

However, the attractive properties mentioned above only apply to the global minimizers of $\mathbf{P}^{(\lambda_1,\lambda_2)}$, which in general will be non-convex, meaning that in practice only local minimizers to the problem can be found using common incremental machine learning algorithms. Thus, we seek conditions on the structure of the learned controller which ensure that the optimization problem is convex. Specifically, we analyze the case where the learned portion of the controller is of the form

$$\tilde{u}(x, \theta) = \sum_{k=1}^{p} \theta_k u_k(x), \tag{11}$$

where $\{u_k\}_{k=1}^p$ is a set of features.

Theorem 2. Suppose that the learned augmentation in (7) is of the form (11), and that the set $\{u_k\}_{k=1}^p$ is linearly independent. Then $\mathbf{P}^{(\lambda_1,\lambda_2)}$ is strongly convex.

Due to space constraints we omit the proof of Theorem 2, as it closely follows the steps in the proof of Lemma 2 in Westenbroek et al. (2020).

Many well-known bases such as radial basis functions (Sanner and Slotine, 1992) or polynomials can be used to recover any continuous function up to a desired degree of accuracy by including enough terms in the expansion. It is an important matter for future work to include these methods in our framework, as it would enable users to design networks for the learned controller which are guaranteed to be able to satisfy the CLF and CBF constraints to a desired degree of accuracy. However, function approximation schemes of the form (11) may require a prohibitive number of bases elements to ensure that the desired function is accurately reconstructed in high dimensions. Thus, in practice, more compact function approximators such as feed-forward neural networks must be used in high dimensions. Unfortunately, such networks generally lead to non-convexities in $\mathbf{P}^{(\lambda_1,\lambda_2)}$.

Numerical Implementation via RL: In practice, our method uses finite difference approximations to $\dot{h}$ and $\dot{V}$ to compute the terms in Δ_1 and Δ_2, and then solves the resulting approximations to $\mathbf{P}^{(\lambda_1,\lambda_2)}$ using standard model-free reinforcement learning algorithms.

Specifically, we will assume that during the learning process the learned controller is sampled every $\Delta t > 0$ seconds, and will let $t_k = k\Delta t$ for $k \in \mathbb{N}$ denote the sampling instances. When the control $\hat{u}(x(t_k), \theta)$ is applied over the interval $[t_k, t_{k+1}]$ we have

$$\Delta_1(x(t_k)), \theta) = \underbrace{-\frac{h(x(t_{k+1})) - h(x(t_k))}{\Delta t} - \alpha(h(x(t_k)))}_{=:\tilde{\Delta}_1(x,\theta)} + O(\Delta t^2),$$

$$\Delta_2(x(t_k)), \theta) = \underbrace{\frac{V(x(t_{k+1})) - V(x(t_k))}{\Delta t} + \sigma(x(t_k))}_{=:\tilde{\Delta}_2(x,\theta)} + O(\Delta t^2).$$

Thus, for small $\Delta t > 0$ we approximate $L^{(\lambda_1,\lambda_2)}$ with

$$\tilde{L}^{(\lambda_1,\lambda_2)}(x, \theta) = \|\hat{u}(x, \theta)\|_2^2 + \lambda_1 H(\tilde{\Delta}_1(x, \theta)) + \lambda_2 H(\tilde{\Delta}_2(x, \theta))$$

and define the following reinforcement learning problem:

$$\tilde{\mathbf{P}}^{(\lambda_1,\lambda_2)}\colon \min_{\theta\in\Theta} E_{x_0\sim X}\left[\sum_{k=0}^{N} \tilde{L}^{(\lambda_1,\lambda_2)}(x_k, \theta)\right] \tag{12}$$
$$\text{s.t. } x_{k+1} = x_k + \textstyle\int_{t_k}^{t_{k+1}} [f(x(t)) + g(x(t))\hat{u}(x_k, \theta)]\, dt$$

Here, $N \in \{1, 2, \dots\}$ is the length of the rollout for each experiment on the plant. Note that this is a standard form for reinforcement learning problems, which can be solved using any off-the-shelf algorithm. Future work will seek to provide correctness guarantees when specific learning algorithms are used to solve these approximations.

4. SIMULATIONS

Due to space constraints, some details of the examples are excluded but can be found in Westenbroek et al. (2021).

Double Pendulum With Safety Constraint: We first apply the learning framework to the double pendulum in Figure 1 with two degrees of freedom $q = (\theta_1, \theta_2) \in \mathbb{R}^2$ and inputs $u = (\tau_1, \tau_2) \in \mathbb{R}^2$, where τ_i is a torque applied at the joints. The Lagrangian dynamics obey

$$M(q)\ddot{q} + \Gamma(q, \dot{q}) = Bu,$$

where $M(q)$ is the mass matrix and $\Gamma(q, \dot{q})$ collects the gravity and Coriolis terms. The overall state of the system is $x = (\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2) \in \mathbb{R}^4$.

The control objective is to stabilize the system to the origin, while ensuring that the y-position of the end-effector does not dip below the constraint depicted in Figure 1. In Figure 1 the origin corresponds to both arms pointing directly to the right. To guide the system towards the origin, the method from Ames et al. (2014) is used to design a CLF of the form $V(x) = x^T P x$. We then design a CBF which ensures satisfaction of the safety constraint using the method of *exponential control barrier functions* (ECBFs) described in Nguyen and Sreenath (2016).

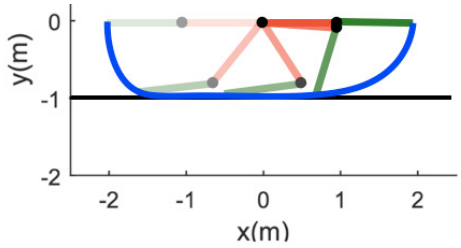

Fig. 1. A trace of a trajectory for the double pendulum under the influence of the learned controller. The horizontal black line represents the safety constraint, while the blue curve traces the end-effector.

To set up the learning problem, we vary the dynamics parameters of the model (mass and length of arms) by 50 percent between the 'true' system dynamics and the nominal model used by the system designer. The learned controller is composed of a linear combination of 300 Gaussian radial basis functions distributed randomly throughout the state-space. We solve the reinforcement learning problem (12) with a rollout length of $N = 1$, penalty parameters $\lambda_1 = 1000$ and $\lambda_2 = 100$ and step-length of $\Delta t = 0.05$s. The Soft Actor Critic (SAC) algorithm from Haarnoja et al. (2018) is used to solve the problem. Figure 1 displays the performance of the learned controller after only 800 samples are collected, which corresponds to 40 seconds of data. The controller was tested from 20 initial conditions, maintaining safety and stability in each scenario.

Safe Bipedal Locomotion on Stepping Stones: We will now apply the presented method to the Hybrid Zero Dynamics (HZD) framework in order to learn an efficient, stable and safe walking controller for a bipedal robot walking on a discrete terrain of randomly spaced stepping stones. The robot is modelled as a hybrid system with impulse effects, as done in Ames et al. (2014):

$$\Sigma : \begin{cases} \dot{\eta} = f(\eta, z) + g(\eta, z)u, & \\ \dot{z} = p(\eta, z) & \text{when } (\eta, z) \notin \mathcal{S}, \\ \eta^+ = \Delta_X\left(\eta^-, z^-\right), & \\ z^+ = \Delta_Z\left(\eta^-, z^-\right) & \text{when } (\eta, z) \in \mathcal{S}, \end{cases} \tag{13}$$

where $\eta \in \mathcal{X} \subset \mathbb{R}^{n_a}$ are the actuated states, $z \in \mathcal{Z} \subset \mathbb{R}^{n_u}$ the unactuated states, and $u \in \mathcal{U} \subseteq \mathbb{R}^m$ the control inputs. This model assumes alternating single support phases, where the swing foot is off the ground and the stance foot remains at a fixed point. 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. In (13), $\eta^+ \in \mathcal{X}$ and $z^+ \in \mathcal{Z}$ are the post-impact states, while $\eta^- \in \mathcal{X}$ and $z^- \in \mathcal{Z}$ are the pre-impact states.

The method of Hybrid Zero Dynamics (HZD) aims to drive the actuated states to zero thereby constraining the system to evolve on a lower dimensional zero dynamics manifold $\Psi = \{(\eta, z) \in \mathcal{X} \times \mathcal{Z} \colon \eta = 0\}$, which contains a stable walking gait for the model. As in Ames et al. (2014), the system can be stabilized to this surface using feedback linearization to construct a CLF for the actuated coordinates. Following the method in Nguyen and Sreenath (2015), we also design a CBF which takes in the relative distance between the current and subsequent stepping stones and forces the robot to step down on the next stepping stone during each impact event. Both of these functions are only used to constrain the evolution of the continuous dynamics, but are constructed so as to maintain safe, stable walking for the full hybrid dynamics. Because of this, we can directly apply our framework to overcome model uncertainty in the continuous dynamics.

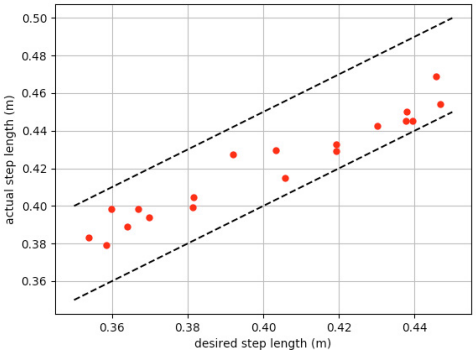

Fig. 2. Plot of the desired step length vs actual step length achieved by the learned controller for the walking simulation. The black dashed lines indicate the necessary step length constraint required to successfully walk over stepping stones.

To set up the learning problem, model uncertainty is introduced by scaling the mass and inertia of each of the robot's links to be three times those of the nominal model. The learned policy takes the form of a neural network with two hidden layers of size 400×300, and *tanh* activation functions. The training data consists of rollouts of 2 consecutive walking steps with randomly perturbed initial conditions and desired step lengths l_d sampled uniformly from $\mathcal{L} := [0.35, 0.45]m$. We again use SAC to train the policy, with a time step of $\Delta t = 1/1000s$ for numerical simulations. The training process converges in about 200,000 time steps, corresponding to about 3 minutes and 20 seconds of data.

The trained policy is tested on 100 simulations of 10 walking steps each, with desired step lengths uniformly sampled from $\mathcal{L}$. The robot only has knowledge of the position of the next stepping stone. A simulation is considered as a failure if the robot fails to land on any of the desired stepping stones, or if it losses stability and falls. Out of the 100 simulations, 93 were successful using the learned controller, while only 26 simulations were successful with the nominal controller without the learning component. This ability of the learned controller to adapt to different required step lengths is clearly reflected in Figure 2.

5. DISCUSSION AND CONCLUSION

In the last section we have shown that our method can learn safe, stabilizing controllers for systems with high model uncertainty with only seconds or a few minutes of training data. However, there are some limitations of our approach. First, as expected by the use of reinforcement learning algorithms, fine tuning of the learning hyperparameters can be time consuming, and in order to get the results shown in this paper an extensive search had to be conducted. It will be an important matter for future work to provide algorithm-dependent guarantees which characterize the solutions obtained by specific methods when solving (approximations to) $\mathbf{P}^{(\lambda_1, \lambda_2)}$. Future work

will also seek to characterize the effects of input saturation on our theoretical guarantees.

Appendix A. PROOF OF THEOREM 1

The overall loss can be written as

$$\mathbb{E}_{x\sim X} L^{(\lambda_1,\lambda_2)}(x,\theta) = M_u(\theta) + \lambda_1 M_1(\theta) + \lambda_2 M_2(\theta).$$

For convenience we write

$$\overline{M}_u = \max_{\theta\in\Theta} M_u(\theta) \qquad \underline{M}_u = \min_{\theta\in\Theta} M_u(\theta)$$
$$\overline{M}_1 = \max_{\theta\in\Theta} M_1(\theta) \qquad \underline{M}_1 = \min_{\theta\in\Theta} M_1(\theta)$$
$$\overline{M}_2 = \max_{\theta\in\Theta} M_2(\theta) \qquad \underline{M}_2 = \min_{\theta\in\Theta} M_2(\theta)$$

We first demonstrate that there exists $C_1, C_2 \geq 0$ such that if $\lambda_1 \geq \frac{1}{\varepsilon_1} C_1\lambda_2 + \frac{1}{\varepsilon_1} C_2$ then for each global optimizer $\theta^* \in \Theta$ of $\mathbf{P}^{(\lambda_1,\lambda_2)}$ we must have $\theta^* \in \Theta_{\varepsilon_1}$. To show this consider two points $\theta_1 \in \Theta_0$ and $\theta_2 \notin \Theta_{\varepsilon_1}$. Let $L_k = \mathbb{E}_{x\sim X} L^{(\lambda_1,\lambda_2)}(x,\theta_k)$ for $k \in \{1,2\}$. We have that:

$$L_1 \leq \overline{M}_u + \lambda_2 \overline{M}_2 \quad \text{and} \quad \underline{M}_u + \lambda_1\varepsilon_1 + \lambda_2 \underline{M}_2 \leq L_2.$$

Here, the first inequality follows from the fact that $M_1(\theta_1) = 0$ and the second inequality follows from the fact that $M_1(\theta_2) \geq \varepsilon_1$. Combining the inequalities yields:

$$L_1 \;\leq\; \left(\overline{M}_u - \underline{M}_u\right) \;-\; \lambda_1\varepsilon_1 \;+\; \lambda_2\left(\overline{M}_2 - \underline{M}_2\right) \;+\; L_2$$

Thus, we see that if we set $\lambda_1 > \frac{1}{\varepsilon_1} C_1\lambda_2 + \frac{1}{\varepsilon_1} C_2$ with $C_1 = \overline{M}_2 - \underline{M}_2$ and $C_2 = \overline{M}_u - \underline{M}_u$, then we must have that $L_1 < L_2$. Thus, any $\theta_2 \notin \Theta_{\varepsilon_1}$ cannot be a global minimizer if we choose the constants $C_1, C_2 \geq 0$ as above.

Next, we demonstrate that when we fix $\lambda_1 \geq \frac{1}{\varepsilon_1} C_1\lambda_2 + \frac{1}{\varepsilon_1} C_2$ as we vary λ_2, then there exists $C_3 \geq 0$ such that if we choose $\lambda_2 \geq \frac{1}{\varepsilon_2} C_3$ then any global optimizer θ^* of $\mathbf{P}^{(\lambda_1,\lambda_2)}$ must lie in $\Theta_{\varepsilon_1,\varepsilon_2}$. As established above, we already know that all optimizers of $\mathbf{P}^{(\lambda_1,\lambda_2)}$ must lie in Θ_{ε_1} in this case. Thus, we will now consider the two points $\theta_3 \in \Theta_{0,0}$ and $\theta_4 \in \left\{\theta \in \Theta_{\varepsilon_1} : M_2(\theta) > \tilde{M}_2 + \varepsilon_2\right\}$, with $\tilde{M}_2$ defined as in (10), so that θ_4 satisfies the desired tolerance for the CBF constraint but not the desired tolerance for the CLF constraint. Again let $L_k = \mathbb{E}_{x\sim X} L^{(\lambda_1,\lambda_2)}(x,\theta_k)$ for $k \in \{3,4\}$. We then have that

$$L_3 \leq \overline{M}_u + \lambda_2 \tilde{M}_2 \quad \text{and} \quad \underline{M}_u + \lambda_2(\tilde{M}_2 + \varepsilon_2) \leq L_4$$

where we have used the fact that $M_1(\theta_3) = 0$, $M_2(\theta_3) = \tilde{M}_2$ and $M_2(\theta_4) \geq \tilde{M}_2 + \varepsilon_2$. Again combining the inequalities and rearranging terms we see that

$$L_3 \leq \left(\overline{M}_u - \underline{M}_u\right) - \lambda_2\varepsilon_2 + L_4.$$

Thus, we see that if we select $C_3 = \overline{M}_u - \underline{M}_u$ and put $\lambda_2 \geq \frac{1}{\varepsilon_2} C_3$ then it must be the case that $L_3 < L_4$ so that θ_4 is not a minimizer. Thus, we see that if we choose $\lambda_1 \geq \frac{1}{\varepsilon_1} C_1\lambda_2 + \frac{1}{\varepsilon_1} C_2$ and $\lambda_2 \geq \frac{1}{\varepsilon_2} C_3$ with all of the constants chosen as above then all minimizers of $\mathbf{P}^{(\lambda_1,\lambda_2)}$ must lie in $\Theta_{\varepsilon_1,\varepsilon_2}$, as desired.

REFERENCES

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

Ames, A.D., Xu, X., Grizzle, J.W., and Tabuada, P. (2017). Control barrier function based quadratic programs for safety critical systems. *IEEE Transactions on Automatic Control*, 62(8), 3861–3876.

Ames, A.D., Galloway, K., Sreenath, K., and Grizzle, J.W. (2014). Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics. *IEEE Transactions on Automatic Control*, 59(4), 876–891.

Berkenkamp, F., Turchetta, M., Schoellig, A., and Krause, A. (2017). Safe model-based reinforcement learning with stability guarantees. In *NeurIPS*, 908–918.

Choi, J., Castañeda, F., Tomlin, C., and Sreenath, K. (2020). Reinforcement Learning for Safety-Critical Control under Model Uncertainty, using Control Lyapunov Functions and Control Barrier Functions. In *Robotics: Science and Systems*. Corvalis, OR.

Grizzle, J.W., Chevallereau, C., Sinnet, R.W., and Ames, A.D. (2014). Models, feedback control, and open problems of 3d bipedal robotic walking. *Automatica*, 50(8), 1955–1988.

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

Hwangbo, J., Sa, I., Siegwart, R., and Hutter, M. (2017). Control of a quadrotor with reinforcement learning. *Robotics and Auto. Letters*, 2(4), 2096–2103.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. (2016). End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research*, 17(1), 1334–1373.

Nguyen, Q. and Sreenath, K. (2015). Safety-critical control for dynamical bipedal walking with precise footstep placement. In *IFAC Analysis and Design of Hybrid Systems*. Atlanta, GA.

Nguyen, Q. and Sreenath, K. (2016). Exponential control barrier functions for enforcing high relative-degree safety-critical constraints. In *American Control Conference*, 322–328.

Nguyen, Q. and Sreenath, K. (2021). Robust safety-critical control for dynamic robotics. *IEEE Transactions on Automatic Control*.

Sanner, R.M. and Slotine, J.J. (1992). Gaussian networks for direct adaptive control. *IEEE Transactions on Neural Networks*, 3(6), 837–863.

Sastry, S. and Bodson, M. (1989). *Adaptive control: stability, convergence and robustness*. Courier Corporation.

Sontag, E.D. (1989). A 'universal' construction of artstein's theorem on nonlinear stabilization. *Systems and Control Letters*, 13(2), 117 – 123.

Taylor, A., Singletary, A., Yue, Y., and Ames, A. (2020). Learning for safety-critical control with control barrier functions. In *L4DC*, 708–717.

Westenbroek, T., Agrawal, A., Castañeda, F., Sastry, S.S., and Sreenath, K. (2021). Combining model-based design and model-free policy optimization to learn safe, stabilizing controllers. *arXiv preprint*.

Westenbroek, T., Castañeda, F., Agrawal, A., Sastry, S.S., and Sreenath, K. (2020). Learning min-norm stabilizing control laws for systems with unknown dynamics. In *IEEE Conference on Decision and Control*, 737–744.

Xu, X., Tabuada, P., Grizzle, J.W., and Ames, A.D. (2015). Robustness of control barrier functions for safety critical control. *IFAC-PapersOnLine*, 48(27), 54–61.