2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)
Macau, China, November 4-8, 2019

Episodic Learning with Control Lyapunov Functions for Uncertain Robotic Systems*

Andrew J. Taylor[1], Victor D. Dorobantu[1], Hoang M. Le, Yisong Yue, Aaron D. Ames

***Abstract*— Many modern nonlinear control methods aim to endow systems with guaranteed properties, such as stability or safety, and have been successfully applied to the domain of robotics. However, model uncertainty remains a persistent challenge, weakening theoretical guarantees and causing implementation failures on physical systems. This paper develops a machine learning framework centered around Control Lyapunov Functions (CLFs) to adapt to parametric uncertainty and unmodeled dynamics in general robotic systems. Our proposed method proceeds by iteratively updating estimates of Lyapunov function derivatives and improving controllers, ultimately yielding a stabilizing quadratic program model-based controller. We validate our approach on a planar Segway simulation, demonstrating substantial performance improvements by iteratively refining on a base model-free controller.**

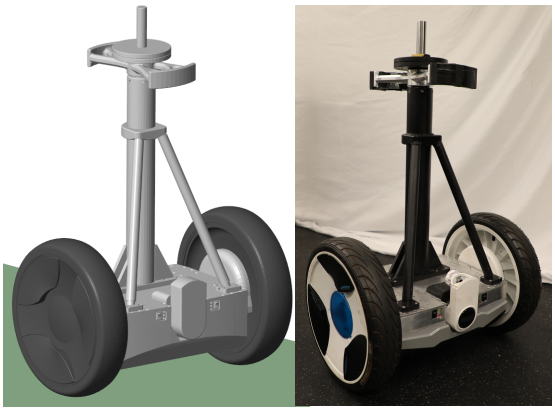

Fig. 1. CAD model & physical system, a modified Ninebot Segway.

I. INTRODUCTION

The use of Control Lyapunov Functions (CLFs) [4], [38] for nonlinear control of robotic systems is becoming increasingly popular [26], [17], [29], often utilizing quadratic program (QP) controllers [2], [1], [17]. While effective, one major challenge is the need for extensive tuning, which is largely due to modeling deficiencies such as parametric error and unmodeled dynamics (cf. [26]). While there has been much research in developing robust control methods that maintain stability under uncertainty (e.g., via input-to-state stability [39]) or in adapting to limited forms of uncertainty (e.g., adaptive control [23], [20]), relatively little work has been done on systematically reducing uncertainty while maintaining stability for general function classes of models.

We take a machine learning approach to address the above limitations. Learning-based approaches have already shown great promise for controlling imperfectly modeled robotic platforms [22], [35]. Successful learning-based approaches have typically focused on learning model-based uncertainty [5], [8], [7], [37], or direct model-free controller design [25], [36], [14], [42], [24].

We are particularly interested in learning-based approaches that guarantee Lyapunov stability [21]. From that perspective, the bulk of previous work has focused on using learning to construct a Lyapunov function [31], [12], [30], or to assess the region of attraction for a Lyapunov function [9], [6]. One limitation of previous work is that learning is conducted over the full-dimensional state space, which can be data inefficient. We instead constructively prescribe a CLF, and focus on learning only the necessary information to choose control inputs that achieve the associated stability guarantees, which can be much lower-dimensional.

One challenge in developing learning-based methods for controller improvement is how best to collect training data that accurately reflects the desired operating environment and control goals. In particular, exhaustive data collection typically scales exponentially with dimensionality of the joint state and control output space, and so should be avoided. But first pre-collecting data upfront can lead to poor performance as downstream control behavior may enter states that are not present in the pre-collected training data. We will leverage episodic learning approaches such as Dataset Aggregation (DAgger) [33] to address these challenges in a data-efficient manner, and lead to iteratively refined controllers.

In this paper we present a novel episodic learning approach that utilizes CLFs to iteratively improve controller design and achieve Lyapunov stability. To the best of our knowledge, our approach is the first that combines CLFs with general supervised learning (e.g., including deep learning) in a mathematically integrated way. Another distinctive aspect is that our approach performs learning on the projection of state dynamics onto the CLF time derivative, which can be much lower dimensional than learning the full state dynamics or the region of attraction.

Our paper is organized as follows. Section II reviews input-output feedback linearization focused on constructing CLFs for unconstrained robotic systems. Section III discusses model uncertainty of a general robotic system and establishes assumptions on the structure of this uncertainty.

[1]Both authors contributed equally.

All authors are with the Department of Computing and Mathematical Sciences, California Institute of Technology, Pasadena, CA 91125, USA ajtaylor@caltech.edu, vdoroban@caltech.edu, hmle@caltech.edu, yyue@caltech.edu, ames@caltech.edu

These assumptions allow us to prescribe a CLF for the true system, but leave open the question of how to model its time derivative. Section IV provides an episodic learning approach to iteratively improving a model of the time derivative of the CLF. We also present a variant of optimal CLF-based control that integrates the learned representation. Finally, Section V provides simulation results on a model of a modified Ninebot by Segway E+, seen in Fig. 1. We also provide a Python software package (LyaPy) implementing our experiments and learning framework.[1]

II. PRELIMINARIES ON CLFs

This section provides a brief review of input-output feedback linearization, a control technique which can be used to synthesize a CLF. The resulting CLF will be used to quantify the impact of model uncertainty and specify the learning problem outlined in Section III.

A. Input-Output Linearization

Input-Output (IO) Linearization is a nonlinear control method that creates stable linear dynamics for a selected set of outputs of a system [34]. Outputs encode information such as the position of a floating-based robot or robotic arm end effector as a function of configuration in a way that is useful for designing controllers. Additionally, IO Linearization provides a constructive method for generating Lyapunov functions, a central tool in certifying stability and synthesizing controllers for nonlinear systems.

Consider an affine robotic control system with configuration space $\mathcal{Q} \subseteq \mathbb{R}^n$ and an input space $\mathcal{U} \subseteq \mathbb{R}^m$. Assume $\mathcal{Q}$ is path-connected and non-empty. The dynamics of the system are specified by:

$$\mathbf{D}(\mathbf{q})\ddot{\mathbf{q}} + \underbrace{\mathbf{C}(\mathbf{q},\dot{\mathbf{q}})\dot{\mathbf{q}} + \mathbf{G}(\mathbf{q})}_{\mathbf{H}(\mathbf{q},\dot{\mathbf{q}})} = \mathbf{B}\mathbf{u}, \tag{1}$$

with generalized coordinates $\mathbf{q} \in \mathcal{Q}$, coordinate rates $\dot{\mathbf{q}} \in \mathbb{R}^n$, input $\mathbf{u} \in \mathcal{U}$, inertia matrix $\mathbf{D} : \mathcal{Q} \to \mathbb{S}^n_{++}$, centrifugal and Coriolis terms $\mathbf{C} : \mathcal{Q} \times \mathbb{R}^n \to \mathbb{R}^{n \times n}$, gravitational forces $\mathbf{G} : \mathcal{Q} \to \mathbb{R}^n$, and static actuation matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$. Here $\mathbb{S}^n_{++}$ denotes the set of $n \times n$ symmetric positive definite matrices. Define twice-differentiable outputs $\mathbf{y} : \mathcal{Q} \to \mathbb{R}^k$, with $k \leq m$, and assume each output has relative degree 2 on some domain $\mathcal{R} \subseteq \mathcal{Q}$ (see [34] for details). Intuitively, the relative degree assumption implies that no configuration in $\mathcal{R}$ results in an inability to actuate the system. Consider the time interval $\mathcal{I} = [t_0, t_f]$ for initial and final times t_0, t_f and define twice-differentiable time-dependent desired outputs $\mathbf{y}_d : \mathcal{I} \to \mathbb{R}^k$ with $\mathbf{r}(t) = \begin{bmatrix}\mathbf{y}_d(t)^\top & \dot{\mathbf{y}}_d(t)^\top\end{bmatrix}^\top$. The error between actual and desired outputs (referred to as virtual constraints [45]) yields the dynamic system:

$$\frac{\mathrm{d}}{\mathrm{d}t}\begin{bmatrix}\mathbf{y}(\mathbf{q}) - \mathbf{y}_d(t) \\ \dot{\mathbf{y}}(\mathbf{q},\dot{\mathbf{q}}) - \dot{\mathbf{y}}_d(t)\end{bmatrix} = \overbrace{\begin{bmatrix}\frac{\partial \mathbf{y}}{\partial \mathbf{q}}\dot{\mathbf{q}} \\ \frac{\partial \dot{\mathbf{y}}}{\partial \mathbf{q}}\dot{\mathbf{q}} - \frac{\partial \mathbf{y}}{\partial \mathbf{q}}\mathbf{D}(\mathbf{q})^{-1}\mathbf{H}(\mathbf{q},\dot{\mathbf{q}})\end{bmatrix}}^{\mathbf{f}(\mathbf{q},\dot{\mathbf{q}})} - \underbrace{\begin{bmatrix}\dot{\mathbf{y}}_d(t) \\ \ddot{\mathbf{y}}_d(t)\end{bmatrix}}_{\dot{\mathbf{r}}(t)} + \underbrace{\begin{bmatrix}\mathbf{0}_{k\times m} \\ \frac{\partial \mathbf{y}}{\partial \mathbf{q}}\mathbf{D}(\mathbf{q})^{-1}\mathbf{B}\end{bmatrix}}_{\mathbf{g}(\mathbf{q})}\mathbf{u}, \tag{2}$$

noting that $\frac{\partial \dot{\mathbf{y}}}{\partial \dot{\mathbf{q}}} = \frac{\partial \mathbf{y}}{\partial \mathbf{q}}$. For all $\mathbf{q} \in \mathcal{R}$, $\mathbf{g}(\mathbf{q})$ is full rank by the relative degree assumption. Define $\boldsymbol{\eta} : \mathcal{Q} \times \mathbb{R}^n \times \mathcal{I} \to \mathbb{R}^{2k}$, $\widetilde{\mathbf{f}} : \mathcal{Q} \times \mathbb{R}^n \to \mathbb{R}^k$, and $\widetilde{\mathbf{g}} : \mathcal{Q} \to \mathbb{R}^{k \times m}$ as:

$$\boldsymbol{\eta}(\mathbf{q},\dot{\mathbf{q}},t) = \begin{bmatrix}\mathbf{y}(\mathbf{q}) - \mathbf{y}_d(t) \\ \dot{\mathbf{y}}(\mathbf{q},\dot{\mathbf{q}}) - \dot{\mathbf{y}}_d(t)\end{bmatrix} \tag{3}$$

$$\widetilde{\mathbf{f}}(\mathbf{q},\dot{\mathbf{q}}) = \frac{\partial \dot{\mathbf{y}}}{\partial \mathbf{q}}\dot{\mathbf{q}} - \frac{\partial \mathbf{y}}{\partial \mathbf{q}}\mathbf{D}(\mathbf{q})^{-1}\mathbf{H}(\mathbf{q},\dot{\mathbf{q}}) \tag{4}$$

$$\widetilde{\mathbf{g}}(\mathbf{q}) = \frac{\partial \mathbf{y}}{\partial \mathbf{q}}\mathbf{D}(\mathbf{q})^{-1}\mathbf{B}, \tag{5}$$

and assume $\mathcal{U} = \mathbb{R}^m$. The input-output linearizing control law $\mathbf{k}_{IO} : \mathcal{Q} \times \mathbb{R}^n \times \mathcal{I} \to \mathcal{U}$ is specified by:

$$\mathbf{k}_{IO}(\mathbf{q},\dot{\mathbf{q}},t) = \widetilde{\mathbf{g}}(\mathbf{q})^\dagger(-\widetilde{\mathbf{f}}(\mathbf{q},\dot{\mathbf{q}}) + \ddot{\mathbf{y}}_d(t) + \boldsymbol{\nu}(\mathbf{q},\dot{\mathbf{q}},t)), \tag{6}$$

with auxiliary input $\boldsymbol{\nu}(\mathbf{q},\dot{\mathbf{q}},t) \in \mathbb{R}^k$ for all $\mathbf{q} \in \mathcal{Q}$, $\dot{\mathbf{q}} \in \mathbb{R}^n$, and $t \in \mathcal{I}$, where $\dagger$ denotes the Moore-Penrose pseudoinverse. Eliminating nonlinear terms, this controller used in (2) generates linear error dynamics of the form:

$$\dot{\boldsymbol{\eta}}(\mathbf{q},\dot{\mathbf{q}},t) = \underbrace{\begin{bmatrix}\mathbf{0}_{k\times k} & \mathbf{I}_{k\times k} \\ \mathbf{0}_{k\times k} & \mathbf{0}_{k\times k}\end{bmatrix}}_{\mathbf{F}}\boldsymbol{\eta}(\mathbf{q},\dot{\mathbf{q}},t) + \underbrace{\begin{bmatrix}\mathbf{0}_{k\times k} \\ \mathbf{I}_{k\times k}\end{bmatrix}}_{\mathbf{G}}\boldsymbol{\nu}(\mathbf{q},\dot{\mathbf{q}},t), \tag{7}$$

where $(\mathbf{F},\mathbf{G})$ are a controllable pair. Defining gain matrix $\mathbf{K} = \begin{bmatrix}\mathbf{K}_p & \mathbf{K}_d\end{bmatrix}$ where $\mathbf{K}_p$, $\mathbf{K}_d \in \mathbb{S}^k_{++}$, the auxiliary control input $\boldsymbol{\nu}(\mathbf{q},\dot{\mathbf{q}},t) = -\mathbf{K}\boldsymbol{\eta}(\mathbf{q},\dot{\mathbf{q}},t)$ induces error dynamics:

$$\dot{\boldsymbol{\eta}}(\mathbf{q},\dot{\mathbf{q}},t) = \mathbf{A}_{cl}\boldsymbol{\eta}(\mathbf{q},\dot{\mathbf{q}},t), \tag{8}$$

where $\mathbf{A}_{cl} = \mathbf{F} - \mathbf{G}\mathbf{K}$ is Hurwitz. This implies the desired output trajectory $\mathbf{y}_d$ is exponentially stable, allowing us to construct a Lyapunov function for the system using converse theorems found in [21]. With $\mathbf{A}_{cl}$ Hurwitz, for any $\mathbf{Q} \in \mathbb{S}^{2k}_{++}$, there exists a unique $\mathbf{P} \in \mathbb{S}^{2k}_{++}$ such that the Continuous Time Lyapunov Equation (CTLE):

$$\mathbf{A}_{cl}^\top\mathbf{P} + \mathbf{P}\mathbf{A}_{cl} = -\mathbf{Q}, \tag{9}$$

is satisfied. Let $\mathcal{C} = \{\boldsymbol{\eta}(\mathbf{q},\dot{\mathbf{q}},t) : (\mathbf{q},\dot{\mathbf{q}}) \in \mathcal{R} \times \mathbb{R}^n, t \in \mathcal{I}\}$. Then $V(\boldsymbol{\eta}) = \boldsymbol{\eta}^\top\mathbf{P}\boldsymbol{\eta}$, implicitly a function of $\mathbf{q}$, $\dot{\mathbf{q}}$, and t, is a Lyapunov function certifying exponential stability of (8) on $\mathcal{C}$ satisfying:

$$\begin{aligned}\lambda_{\min}(\mathbf{P})\|\boldsymbol{\eta}\|_2^2 \leq V(\boldsymbol{\eta}) \leq \lambda_{\max}(\mathbf{P})\|\boldsymbol{\eta}\|_2^2 \\ \dot{V}(\boldsymbol{\eta}) \leq -\lambda_{\min}(\mathbf{Q})\|\boldsymbol{\eta}\|_2^2,\end{aligned} \tag{10}$$

for all $\boldsymbol{\eta} \in \mathcal{C}$. Here $\lambda_{\min}(\cdot)$ and $\lambda_{\max}(\cdot)$ denote the minimum and maximum eigenvalues of a symmetric matrix,

[1] https://github.com/vdorobantu/lyapy

respectively. A similar Lyapunov function can be constructed directly from (7) using the Continuous Algebraic Riccati Equation (CARE) [1], [21].

B. Control Lyapunov Functions

Lyapunov functions encode information about the dynamics in a low-dimensional representation suitable for learning. The preceding formulation of a Lyapunov function required the choice of the specific control law given in (6) to analyze stability of a closed-loop system. More generally, Control Lyapunov Functions (CLFs) extend this idea to enable synthesis of optimal nonlinear controllers. Let $\mathcal{C} \subseteq \mathbb{R}^{2k}$. A function $V : \mathbb{R}^{2k} \to \mathbb{R}_+$ is a Control Lyapunov Function (CLF) for (1) on $\mathcal{C}$ certifying exponential stability if there exist constants $c_1, c_2, c_3 > 0$ such that:

$$\begin{aligned} c_1\|\boldsymbol{\eta}\|_2^2 \leq V(\boldsymbol{\eta}) \leq c_2\|\boldsymbol{\eta}\|_2^2 \\ \inf_{\mathbf{u}\in\mathcal{U}} \dot{V}(\boldsymbol{\eta}, \mathbf{u}) \leq -c_3\|\boldsymbol{\eta}\|_2^2, \end{aligned} \tag{11}$$

for all $\boldsymbol{\eta} \in \mathcal{C}$. We see that the previously constructed Lyapunov function satisfying (10) satisfies (11) by choosing the control law specified in (6). In the absence of a specific control law, we may write the CLF time derivative as:

$$\dot{V}(\boldsymbol{\eta}, \mathbf{u}) = \frac{\partial V}{\partial \boldsymbol{\eta}}\dot{\boldsymbol{\eta}} = \frac{\partial V}{\partial \boldsymbol{\eta}}(\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}) - \dot{\mathbf{r}}(t) + \mathbf{g}(\mathbf{q})\mathbf{u}). \tag{12}$$

Dynamic information directly appears within the scalar function $\dot{V}$. Also note that $\dot{V}$ is affine in $\mathbf{u}$, leading to a QP based control law $\mathbf{k}_{QP} : \mathcal{Q} \times \mathbb{R}^n \times \mathcal{I} \to \mathcal{U}$ given by:

$$\begin{aligned} \mathbf{k}_{QP}(\mathbf{q}, \dot{\mathbf{q}}, t) = \underset{\mathbf{u}\in\mathcal{U}}{\arg\min} \quad & \frac{1}{2}\mathbf{u}^\top\mathbf{M}\mathbf{u} + \mathbf{s}^\top\mathbf{u} + r \\ \text{s.t.} \quad & \dot{V}(\boldsymbol{\eta}, \mathbf{u}) \leq -c_3\|\boldsymbol{\eta}\|_2^2, \end{aligned} \tag{13}$$

for $\mathbf{M} \in \mathbb{S}_+^m$, $\mathbf{s} \in \mathbb{R}^m$, and $r \in \mathbb{R}$, provided $\mathcal{U}$ is a polyhedron. Here $\mathbb{S}_+^m$ denotes the set of $m \times m$ symmetric positive semi-definite matrices.

III. UNCERTAINTY MODELS & LEARNING

This section defines the class of model uncertainty we consider in this work and investigates its impact on the control system, and concludes with motivation for a data-driven approach to mitigate this impact.

A. Uncertainty Modeling Assumptions

As defined in Section II, we consider affine robotic control systems that evolve under dynamics described by (1). In practice, we do not know the dynamics of the system exactly, and instead develop our control systems using the estimated model:

$$\widehat{\mathbf{D}}(\mathbf{q})\ddot{\mathbf{q}} + \underbrace{\widehat{\mathbf{C}}(\mathbf{q}, \dot{\mathbf{q}})\dot{\mathbf{q}} + \widehat{\mathbf{G}}(\mathbf{q})}_{\widehat{\mathbf{H}}(\mathbf{q},\dot{\mathbf{q}})} = \widehat{\mathbf{B}}\mathbf{u}. \tag{14}$$

We assume the estimated model (14) satisfies the relative degree condition on the domain $\mathcal{R}$, and thus may use the method of input-output linearization to produce a Control Lyapunov Function (CLF), V, for the system. Using the definitions established in (2) in conjunction with the estimated model, we see that true system evolves as:

$$\begin{aligned} \dot{\boldsymbol{\eta}} = \quad & \widehat{\mathbf{f}}(\mathbf{q}, \dot{\mathbf{q}}) - \dot{\mathbf{r}}(t) + \widehat{\mathbf{g}}(\mathbf{q})\mathbf{u} \\ & + \underbrace{(\mathbf{g}(\mathbf{q}) - \widehat{\mathbf{g}}(\mathbf{q}))}_{\mathbf{A}(\mathbf{q})}\mathbf{u} + \underbrace{\mathbf{f}(\mathbf{q}, \dot{\mathbf{q}}) - \widehat{\mathbf{f}}(\mathbf{q}, \dot{\mathbf{q}})}_{\mathbf{b}(\mathbf{q},\dot{\mathbf{q}})}. \end{aligned} \tag{15}$$

We note the following features of modeling uncertainty in this fashion:

- Uncertainty is allowed to enter the system dynamics via parametric error as well as through completely unmodeled dynamics. In particular, the function $\mathbf{H}$ can capture a wide variety of nonlinear behavior and only needs to be Lipschitz continuous.
- This formulation explicitly allows uncertainty in how the input is introduced into the dynamics via uncertainty in the inertia matrix $\mathbf{D}$ and static actuation matrix $\mathbf{B}$. This definition of uncertainty is also compatible with a dynamic actuation matrix $\mathbf{B} : \mathcal{Q} \times \mathbb{R}^n \to \mathbb{R}^{n\times m}$ given proper assumptions on the relative degree of the system.

Given this formulation of our uncertainty, we make the following assumptions of the true dynamics:

Assumption 1. *The true system is assumed to be deterministic, time invariant, and affine in the control input.*

Assumption 2. *The CLF V, formulated for the estimated model, is a CLF for the true system.*

It is sufficient to assume that the true system have relative degree 2 on the domain $\mathcal{R}$ to satisfy Assumption 2. This holds since the true values of $\widetilde{\mathbf{f}}$ and $\widetilde{\mathbf{g}}$, if known, enable choosing control inputs as in (6) that respect the same linear error dynamics (8). Given that V is a CLF for the true system, its time derivative under uncertainty is given by:

$$\begin{aligned} \dot{V}(\boldsymbol{\eta}, \mathbf{u}) = \quad & \overbrace{\frac{\partial V}{\partial \boldsymbol{\eta}}(\widehat{\mathbf{f}}(\mathbf{q}, \dot{\mathbf{q}}) - \dot{\mathbf{r}}(t) + \widehat{\mathbf{g}}(\mathbf{q})\mathbf{u})}^{\widehat{\dot{V}}(\boldsymbol{\eta},\mathbf{u})} \\ & + \underbrace{\frac{\partial V}{\partial \boldsymbol{\eta}}\mathbf{A}(\mathbf{q})}_{\mathbf{a}(\boldsymbol{\eta},\mathbf{q})^\top}\mathbf{u} + \underbrace{\frac{\partial V}{\partial \boldsymbol{\eta}}\mathbf{b}(\mathbf{q}, \dot{\mathbf{q}})}_{b(\boldsymbol{\eta},\mathbf{q},\dot{\mathbf{q}})}, \end{aligned} \tag{16}$$

for all $\boldsymbol{\eta} \in \mathbb{R}^{2k}$ and $\mathbf{u} \in \mathcal{U}$. While V is a CLF for the true system, it is no longer possible to determine if a specific control value will satisfy the derivative condition in (11) due to the unknown components $\mathbf{a}$ and b. Rather than form a new Lyapunov function, we seek to better estimate the Lyapunov function derivative $\dot{V}$ to enable control selection that satisfies the exponential stability requirement. This estimate should be affine in the control input, enabling its use in the controller described in (13). Instead of learning the unknown dynamics terms $\mathbf{A}$ and $\mathbf{b}$, which scale with both the dimension of the configuration space and the number of inputs, we will learn the terms $\mathbf{a}$ and b, which scale only with the number of inputs. In the case of the planar Segway model we simulate, we reduce the number of learned components from

4 to 2 (assuming kinematics are known). These learned representations need to accurately capture the uncertainty over the domain in which the system is desired to evolve to ensure stability during operation.

B. Motivating a Data-Driven Learning Approach

The formulation from (15) and (16) defines a general class of dynamics uncertainty. It is natural to consider a data-driven method to estimate the unknown quantities $\mathbf{a}$ and b over the domain of the system. To motivate our learning-based framework, first consider a simple approach of learning $\mathbf{a}$ and b via supervised regression [19]: we operate the system using some given state-feedback controller to gather data points along the system's evolution and learn a function that approximates $\mathbf{a}$ and b via supervised learning.

Concretely, let $\mathbf{q}_0 \in \mathcal{Q}$ be an initial configuration. An experiment is defined as the evolution of the system over a finite time interval from the initial condition $(\mathbf{q}_0, \mathbf{0})$ using a discrete-time implementation of the given controller. A resulting discrete-time state history is obtained, which is then transformed with Lyapunov function V and finally differentiated numerically to estimate $\dot{V}$ throughout the experiment. This yields a data set comprised of input-output pairs:

$$D = \{((\mathbf{q}_i, \dot{\mathbf{q}}_i, \boldsymbol{\eta}_i, \mathbf{u}_i), \dot{V}_i)\}_{i=1}^N \subseteq (\mathcal{Q} \times \mathbb{R}^n \times \mathbb{R}^{2k} \times \mathcal{U}) \times \mathbb{R}. \tag{17}$$

Consider a class $\mathcal{H}_{\mathbf{a}}$ of nonlinear functions mapping from $\mathbb{R}^{2k} \times \mathcal{Q}$ to $\mathbb{R}^m$ and a class $\mathcal{H}_b$ of nonlinear functions mapping from $\mathbb{R}^{2k} \times \mathcal{Q} \times \mathbb{R}^n$ to $\mathbb{R}$. For a given $\widehat{\mathbf{a}} \in \mathcal{H}_{\mathbf{a}}$ and $\widehat{b} \in \mathcal{H}_b$, define $\widehat{\dot{W}}$ as:

$$\widehat{\dot{W}}(\boldsymbol{\eta}, \mathbf{q}, \dot{\mathbf{q}}, \mathbf{u}) = \widehat{\dot{V}}(\boldsymbol{\eta}, \mathbf{u}) + \widehat{\mathbf{a}}(\boldsymbol{\eta}, \mathbf{q})^\top \mathbf{u} + \widehat{b}(\boldsymbol{\eta}, \mathbf{q}, \dot{\mathbf{q}}), \tag{18}$$

and let $\mathcal{H}$ be the class of all such estimators mapping $\mathbb{R}^{2k} \times \mathcal{Q} \times \mathbb{R}^n \times \mathcal{U}$ to $\mathbb{R}$. Defining a loss function $\mathcal{L} : \mathbb{R} \times \mathbb{R} \to \mathbb{R}_+$, the supervised regression task is then to find a function in $\mathcal{H}$ via empirical risk minimization (ERM):

$$\inf_{\substack{\widehat{\mathbf{a}} \in \mathcal{H}_{\mathbf{a}} \\ \widehat{b} \in \mathcal{H}_b}} \frac{1}{N} \sum_{i=1}^N \mathcal{L}(\widehat{\dot{W}}(\boldsymbol{\eta}_i, \mathbf{q}_i, \dot{\mathbf{q}}_i, \mathbf{u}_i), \dot{V}_i). \tag{19}$$

This experiment protocol can be executed either in simulation or directly on hardware. While being simple to implement, supervised learning critically assumes independently and identically distributed (i.i.d) training data. Each experiment violates this assumption, as the regression target of each data point is coupled with the input data of the next time step. As a consequence, standard supervised learning with sequential, non-i.i.d data collection often leads to error cascades [24].

IV. INTEGRATING EPISODIC LEARNING & CLFs

In this section we present the main contribution of this work: an episodic learning algorithm that captures the uncertainty present in the Lyapunov function derivative in a learned model and utilizes it in a QP based controller.

Algorithm 1 Dataset Aggregation for Control Lyapunov Functions (DaCLyF)

Require: Control Lyapunov Function V, derivative estimate $\widehat{\dot{V}}_0$, model classes $\mathcal{H}_{\mathbf{a}}$ and $\mathcal{H}_b$, loss function $\mathcal{L}$, set of initial configurations $\mathcal{Q}_0$, nominal state-feedback controller $\mathbf{k}_0$, number of experiments T, sequence of trust coefficients $0 \leq w_1 \leq \cdots \leq w_T \leq 1$

$D = \emptyset$ ▷ Initialize data set
for $k = 1, \ldots, T$ **do**
  $(\mathbf{q}_0, \mathbf{0}) \leftarrow \text{sample}(\mathcal{Q}_0 \times \{\mathbf{0}\})$ ▷ Get initial condition
  $D_k \leftarrow \text{experiment}((\mathbf{q}_0, \mathbf{0}), \mathbf{k}_{k-1})$ ▷ Run experiment
  $D \leftarrow D \cup D_k$ ▷ Aggregate data set
  $\widehat{\mathbf{a}}, \widehat{b} \leftarrow \text{ERM}(\mathcal{H}_{\mathbf{a}}, \mathcal{H}_b, \mathcal{L}, D, \widehat{\dot{V}}_0)$ ▷ Fit estimators
  $\widehat{\dot{V}}_k \leftarrow \widehat{\dot{V}}_0 + \widehat{\mathbf{a}}^\top \mathbf{u} + \widehat{b}$ ▷ Update derivative estimator
  $\mathbf{k}_k \leftarrow \mathbf{k}_0 + w_k \cdot \text{augment}(\mathbf{k}_0, \widehat{\dot{V}}_k)$ ▷ Update controller
end for
return $\widehat{\dot{V}}_T, \mathbf{u}_T$

A. Episodic Learning Framework

Episodic learning refers to learning procedures that iteratively alternates between executing an intermediate controller (also known as a roll-out in reinforcement learning [22]), collecting data from that roll-out, and designing a new controller using the newly collected data. Our approach integrates learning $\mathbf{a}$ and b with improving the performance and stability of the control policy $\mathbf{u}$ in such an iterative fashion. First, assume we are given a nominal state-feedback controller $\mathbf{k} : \mathcal{Q} \times \mathbb{R}^n \times \mathcal{I} \to \mathcal{U}$, which may not stabilize the system. With an estimator $\widehat{\dot{W}} \in \mathcal{H}$ as defined in (18), we specify an augmenting controller as:

$$\begin{aligned} \mathbf{k}'(\mathbf{q}, \dot{\mathbf{q}}, t) = \; & \underset{\mathbf{u}' \in \mathbb{R}^m}{\arg\min}\, J(\mathbf{u}') \\ & \text{s.t. } \widehat{\dot{W}}(\boldsymbol{\eta}, \mathbf{q}, \dot{\mathbf{q}}, \mathbf{k}(\mathbf{q}, \dot{\mathbf{q}}, t) + \mathbf{u}') \leq -c_3 \|\boldsymbol{\eta}\|_2^2 \\ & \quad \mathbf{k}(\mathbf{q}, \dot{\mathbf{q}}, t) + \mathbf{u}' \in \mathcal{U}, \end{aligned} \tag{20}$$

where $J : \mathbb{R}^m \to \mathbb{R}$ is any positive semi-definite quadratic cost function. This augmenting control law effectively finds the minimal addition $\mathbf{u}'$ to the input determined by the nominal control law $\mathbf{k}$ such that the sum stabilizes the system; however, stability degrades with error in the estimator $\widehat{\dot{W}}$.

In an effort to reduce the remaining error, we use this new controller to obtain better estimates of $\mathbf{a}$ and b. One option, as seen in Section III-B, is to perform experiments and use conventional supervised regression to update $\widehat{\mathbf{a}}$ and $\widehat{b}$. To overcome the limitations of conventional supervised learning, we leverage reduction techniques: a sequential prediction problem is reduced to a sequence of supervised learning problems over multiple episodes [15], [32]. In particular, in each episode, an experiment generates data using a different controller. The data set is aggregated and a new ERM problem is solved after each episode. Our episodic learning implementation is inspired by the Data Aggregation algorithm (DAgger) [32], with some key differences:

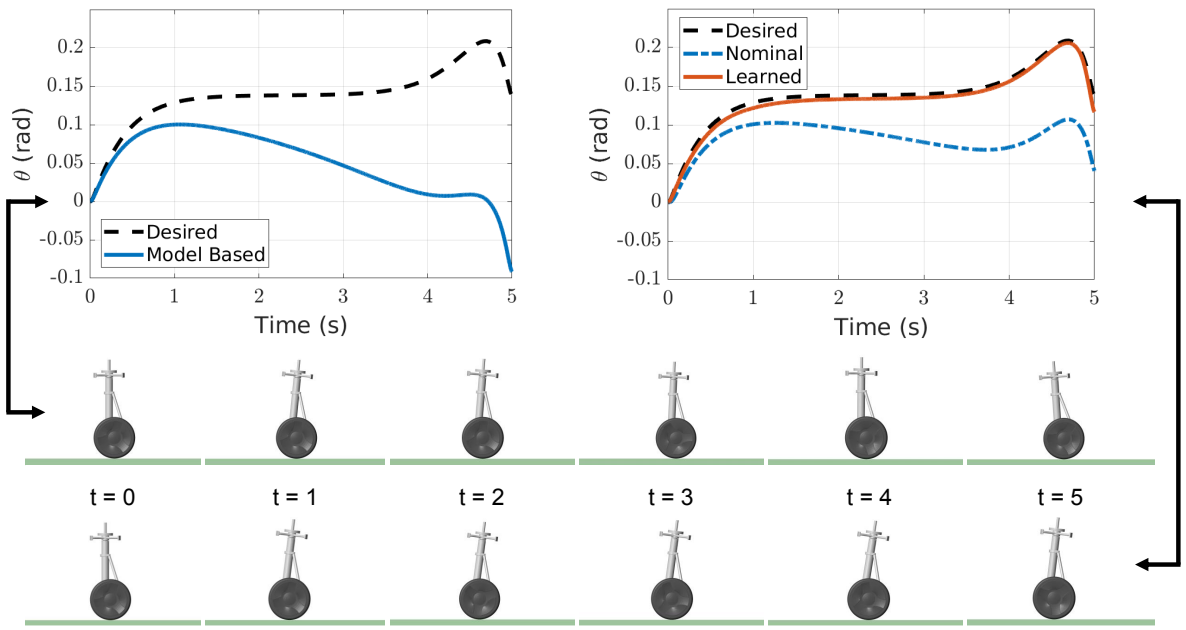

Fig. 2. (Left) Model based QP controller fails to track trajectory. (Right) Improvement in angle tracking of system with augmented controller over nominal PD controller. (Bottom) Corresponding visualizations of state data. Note that Segway is tilted in the incorrect direction at the end of the QP controller simulation, but is correctly aligned during the augmented controller simulation. Video of this animation is found at `https://youtu.be/cB5MY_8vCrQ`.

- DAgger is a model-free policy learning algorithm, which trains a policy directly in each episode using optimal computational oracles. Our algorithm defines a controller indirectly via a CLF to ensure stability.
- The ERM problem is underdetermined, i.e., different approximations $(\widehat{\mathbf{a}}, \widehat{b})$ may achieve similar loss for a given data set while failing to accurately model $\mathbf{a}$ and b. This potentially introduces error in estimating $\dot{V}$ for control inputs not reflected in the training data, and necessitates the use of exploratory control action to constrain the estimators $\widehat{\mathbf{a}}$ and $\widehat{b}$. Such exploration can be achieved by randomly perturbing the controller used in an experiment at each time step. This need for exploration is an analog to the notion of persistent excitation from adaptive systems [28].

Algorithm 1 specifies a method of computing a sequence of Lyapunov function derivative estimates and augmenting controllers. During each episode, the augmenting controller associated with the estimate of the Lyapunov function derivative is scaled by a heuristically chosen factor reflecting trust in the estimate and added to the nominal controller for use in the subsequent experiment. The trust coefficients form a monotonically non-decreasing sequence on the interval $[0, 1]$. Importantly, this experiment need not take place in simulation; the same procedure may be executed directly on hardware. It may be infeasible to choose a specific configuration for an initial condition on a hardware platform; therefore we specify a set of initial configurations $\mathcal{Q}_0 \subseteq \mathcal{Q}$ from which an initial condition may be sampled, potentially randomly. At a high level, this episodic approach makes progress by gathering more data in relevant regions of the state space, such as states close to a target trajectory. This extends the generalizability of the estimator in its use by subsequent controllers, and improves stability results as explored in [43].

B. Additional Controller Details

During augmentation, we specify the controller in (20) by selecting the minimum-norm cost function:

$$J(\mathbf{u}') = \frac{1}{2} \left\| \mathbf{k}(\mathbf{q}, \dot{\mathbf{q}}, t) + \mathbf{u}' \right\|_2^2, \tag{21}$$

for all $\mathbf{u}' \in \mathbb{R}^m$, $\mathbf{q} \in \mathcal{Q}$, $\dot{\mathbf{q}} \in \mathbb{R}^n$, and $t \in \mathcal{I}$. For practical considerations we incorporate the following smoothing term into the cost:

$$R(\mathbf{u}') = \overline{R} \left\| \mathbf{u}' - \mathbf{u}_{\text{prev}} \right\|_2^2,$$

for all $\mathbf{u}' \in \mathbb{R}^m$, where $\mathbf{u}_{\text{prev}} \in \mathbb{R}^m$ is the previously computed augmenting controller and $\overline{R} > 0$. This is done to avoid chatter that may arise from the optimization based nature of the CLF-QP formulation [27].

Note that for this choice of Lyapunov function, the gradient $\frac{\partial V}{\partial \boldsymbol{\eta}}$, and therefore $\mathbf{a}$, approach $\mathbf{0}$ as $\boldsymbol{\eta}$ approaches $\mathbf{0}$, which occurs close to the desired trajectory. While the estimated Lyapunov function derivative may be fit with low absolute error on the data set, the relative error may still be high for states near the desired trajectory. Such relative error causes the optimization problem in (20) to be poorly conditioned near the desired trajectory. We therefore add a slack term $\delta \in \mathbb{R}_+$ to the decision variables, which appears in the inequality constraint [2]. The slack term is additionally

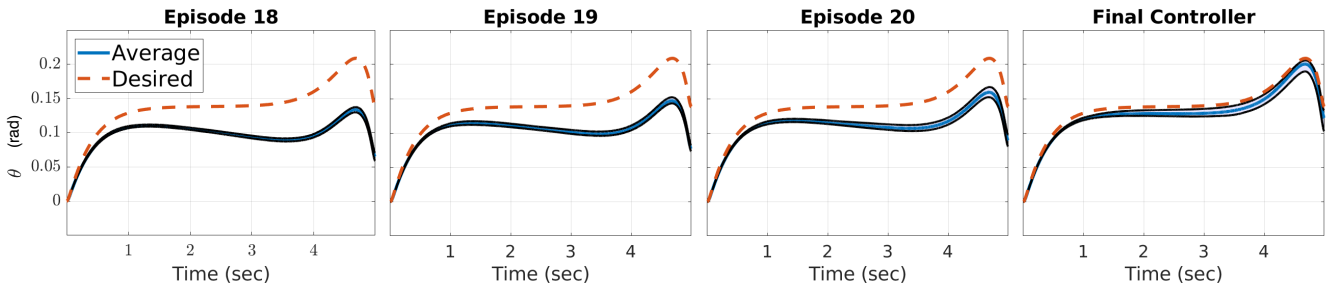

Fig. 3. Augmenting controllers consistently improve trajectory tracking across episodes. 10 instances of the algorithm are executed with the shaded region formed from minimum and maximum angles for each time step within an episode. The corresponding average angle trajectories are also displayed.

incorporated into the cost as:

$$C(\delta) = \frac{1}{2}\overline{C}\left\| \left(\frac{\partial V}{\partial \boldsymbol{\eta}}\widehat{\mathbf{g}}(\mathbf{q})\right)^{\top} + \widehat{\mathbf{a}}(\boldsymbol{\eta}, \mathbf{q}) \right\|_2^2 \delta^2, \tag{22}$$

for all $\delta \in \mathbb{R}_+$, where $\overline{C} > 0$. As states approach the trajectory, the coefficient of the quadratic term decreases and enables relaxation of the exponential stability inequality constraint. In practice this leads to input-to-state stable behavior, described in [40], around the trajectory.

The exploratory control during experiments is naively chosen as additive noise from a centered uniform distribution, with each coordinate drawn i.i.d. The variance is scaled by the norm of the underlying controller to introduce exploration while maintaining a high signal-to-noise ratio.

V. APPLICATION ON SEGWAY PLATFORM

In this section we apply the episodic learning algorithm constructed in Section IV to the Segway platform. In particular, we consider a 4-dimensional planar Segway model based on the simulation model in [18]. The system states consist of horizontal position and velocity, pitch angle, and pitch angle rate. Control is specified as a single voltage input supplied to both motors. The parameters of the model (including mass, inertias, and motor parameters but excluding gravity) are randomly modified by up to 10% of their nominal values and are fixed for the simulations.

We seek to track a pitch angle trajectory[2] generated for the estimated model. The nominal controller is a linear proportional-derivative (PD) controller on angle and angle rate error. 20 experiments are conducted with trust values varying from 0.01 to 0.99 in a sigmoid fashion. The exploratory control is drawn uniformly at random between -20% and 20% of the norm of the underlying controller for an episode for the first 10 episodes. The percentages decay linearly to 0 in the remaining 10 episodes. The model classes selected are sets of two-layer neural networks with ReLU nonlinearities with hidden layer width of 2000 nodes[3]. The inputs to both models are all states and the CLF gradient.

Failure of the controller (13) designed for the estimated model to track the desired trajectory is seen in the left portion of Fig. 2. The baseline PD controller and the augmented controller after 20 experiments can be seen in the right portion Fig. 2. Corresponding visualizations of the Segway states are displayed at the bottom of Fig. 2. The augmented controller exhibits a notable improvement over the model-based and PD controller in tracking the trajectory.

To verify the robustness of the learning algorithm, the 20 experiment process was conducted 10 times. After each experiment the intermediate augmented controller was tested without exploratory perturbations. For the last three experiments and a test of the final augmented controller, the minimum, mean, and maximum angles across all 10 instances are displayed for each time step in Fig. 3. The mean trajectory consistently improves in these later episodes as the trust factor increases. The variation increases but remains small, indicating that the learning problem is robust to randomness in the initialization of the neural networks, in the network training algorithm, and in the noise added during the experiments. The performance of the controller in the earlier episodes displayed negligible variation from the baseline PD controller due to small trust factors.

VI. CONCLUSIONS & FUTURE WORK

We presented an episodic learning framework that directly integrates into an established method of nonlinear control using CLFs. Our method allows for the effects of both parametric error and unmodeled dynamics to be learned from experimental data and incorporated into an QP controller. The success of this approach was demonstrated in simulation on a Segway, showing improvement upon a model estimate based controller.

There are two main interesting directions for future work. First, a more thorough investigation of episodic learning algorithms can yield superior performance as well as learning-theoretic converge guarantees. Other episodic learning approaches to consider include SEARN [13], AggreVaTeD [41], and MoBIL [10], amongst others. Second, our approach can also be applied to learning with other forms of guarantees, such as with Control Barrier Functions (CBFs) [3]. Existing work on learning CBFs are restricted to learning with Gaussian processes [44], [16], [11], and also learn over the full state space rather than over the low-dimensional projection onto the CBF time derivative.

Acknowledgements. This work was supported in part by funding and gifts from DARPA, Intel, PIMCO, and Google.

[2]Trajectory was generated using the GPOPS-II Optimal Control Software
[3]Models were implemented in Keras

REFERENCES

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

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

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

[4] Z. Artstein. Stabilization with relaxed controls. *Nonlinear Analysis: Theory, Methods & Applications*, 7(11):1163–1173, 1983.

[5] T. Beckers, D. Kulić, and S. Hirche. Stable gaussian process based tracking control of euler–lagrange systems. *Automatica*, 103:390–397, 2019.

[6] F. Berkenkamp, R. Moriconi, A. Schoellig, and A. Krause. Safe learning of regions of attraction for uncertain, nonlinear systems with gaussian processes. In *55th Conference on Decision and Control (CDC)*, pages 4661–4666. IEEE, 2016.

[7] F. Berkenkamp and A. Schoellig. Safe and robust learning control with gaussian processes. In *2015 European Control Conference (ECC)*, pages 2496–2501. IEEE, 2015.

[8] F. Berkenkamp, A. Schoellig, and A. Krause. Safe controller optimization for quadrotors with gaussian processes. In *2016 IEEE International Conference on Robotics and Automation (ICRA)*, pages 491–496. IEEE, 2016.

[9] F. Berkenkamp, M. Turchetta, A. Schoellig, and A. Krause. Safe model-based reinforcement learning with stability guarantees. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30*, pages 908–918. Curran Associates, Inc., 2017.

[10] C. Cheng, X. Yan, E. Theodorou, and B. Boots. Accelerating imitation learning with predictive models. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2019.

[11] R. Cheng, G. Orosz, R.M. Murray, and J.W. Burdick. End-to-end safe reinforcement learning through barrier functions for safety-critical continuous control tasks. 2019.

[12] Y. Chow, O. Nachum, E. Duenez-Guzman, and M. Ghavamzadeh. A lyapunov-based approach to safe reinforcement learning. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems 31*, pages 8092–8101. Curran Associates, Inc., 2018.

[13] H. Daumé, J. Langford, and D. Marcu. Search-based structured prediction. *Machine learning*, 75(3):297–325, 2009.

[14] Y. Duan, X. Chen, R. Houthooft, J. Schulman, and P. Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pages 1329–1338, 2016.

[15] D. Ernst, P. Geurts, and L. Wehenkel. Tree-based batch mode reinforcement learning. *Journal of Machine Learning Research*, 6(Apr):503–556, 2005.

[16] J. Fisac, A. Akametalu, M. Zeilinger, S. Kaynama, J. Gillula, and C. Tomlin. A general safety framework for learning-based control in uncertain robotic systems. *IEEE Transactions on Automatic Control*, 2018.

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

[18] T. Gurriet, A. Singletary, J. Reher, L. Ciarletta, E. Feron, and A. Ames. Towards a framework for realizable safety critical control through active set invariance. In *Proceedings of the 9th ACM/IEEE International Conference on Cyber-Physical Systems*, pages 98–106. IEEE Press, 2018.

[19] L. Györfi, M. Kohler, A. Krzyzak, and H. Walk. *A distribution-free theory of nonparametric regression*. Springer Science & Business Media, 2006.

[20] I. Karafyllis, M. Kontorinaki, and M. Krstic. Adaptive control by regulation-triggered batch least-squares estimation of non-observable parameters. *arXiv preprint arXiv:1811.10833*, 2018.

[21] H.K. Khalil. *Nonlinear Systems - 3rd Edition*. PH, Upper Saddle River, NJ, 2002.

[22] J. Kober, J.A. Bagnell, and J. Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013.

[23] M. Krstić and P. Kokotović. Control lyapunov functions for adaptive nonlinear stabilization. *Systems & Control Letters*, 26(1):17–23, 1995.

[24] H. Le, A. Kang, Y. Yue, and P. Carr. Smooth imitation learning for online sequence prediction. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning-Volume 48*, pages 680–688. JMLR. org, 2016.

[25] T. Lillicrap, J. Hunt, A. Pritzel, N. Heess, T. Erez, Y. Tassa, D. Silver, and D. Wierstra. Continuous control with deep reinforcement learning. *arXiv preprint arXiv:1509.02971*, 2015.

[26] W. Ma, S. Kolathaya, E. Ambrose, C. Hubicki, and A. Ames. Bipedal robotic running with durus-2d: Bridging the gap between theory and experiment. In *Proceedings of the 20th International Conference on Hybrid Systems: Computation and Control*, pages 265–274. ACM, 2017.

[27] B. Morris, M. Powell, and A. Ames. Sufficient conditions for the lipschitz continuity of qp-based multi-objective control of humanoid robots. In *52nd IEEE Conference on Decision and Control*, pages 2920–2926. IEEE, 2013.

[28] K. Narendra and A. Annaswamy. Persistent excitation in adaptive systems. *International Journal of Control*, 45(1):127–160, 1987.

[29] Q. Nguyen and K. Sreenath. Optimal robust control for bipedal robots through control lyapunov function based quadratic programs. In *Robotics: Science and Systems*, 2015.

[30] H. Ravanbakhsh and S. Sankaranarayanan. Learning lyapunov (potential) functions from counterexamples and demonstrations. *arXiv preprint arXiv:1705.09619*, 2017.

[31] S. Richards, F. Berkenkamp, and A. Krause. The lyapunov neural network: Adaptive stability certification for safe learning of dynamic systems. *arXiv preprint arXiv:1808.00924*, 2018.

[32] S. Ross, G. Gordon, and D. Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635, 2011.

[33] S. Ross, G. Gordon, and J.A Bagnell. No-regret reductions for imitation learning and structured prediction. *CoRR*, abs/1011.0686, 2010.

[34] S. S. Sastry. *Nonlinear Systems: Analysis, Stability and Control*. Springer, NY, 1999.

[35] S. Schaal and C. Atkeson. Learning control in robotics. *IEEE Robotics & Automation Magazine*, 17(2):20–29, 2010.

[36] J. Schulman, F. Wolski, P. Dhariwal, A. Radford, and O. Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

[37] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.J. Chung. Neural lander: Stable drone landing control using learned dynamics. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2019.

[38] E. Sontag. A 'universal' contruction of Artstein's theorem on nonlinear stabilization. *Systems & Control Letters*, 13:117–123, 1989.

[39] E. Sontag. Input to state stability: Basic concepts and results. In *Nonlinear and optimal control theory*, pages 163–220. Springer, 2008.

[40] E. Sontag and Y. Wang. On characterizations of the input-to-state stability property. *Systems & Control Letters*, 24(5):351–359, 1995.

[41] W. Sun, A. Venkatraman, G. Gordon, B. Boots, and A. Bagnell. Deeply aggrevated: Differentiable imitation learning for sequential prediction. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, pages 3309–3318. JMLR. org, 2017.

[42] J. Tan, T. Zhang, E. Coumans, A. Iscen, Y. Bai, D. Hafner, S. Bohez, and V. Vanhoucke. Sim-to-real: Learning agile locomotion for quadruped robots. *arXiv preprint arXiv:1804.10332*, 2018.

[43] A. Taylor, V. Dorobantu, M. Krishnamoorthy, H. Le, Y. Yue, and A. Ames. A control lyapunov perspective on episodic learning via projection to state stability. *58th Conference on Decision & Control (CDC)*, 2019.

[44] L. Wang, E. Theodorou, and M. Egerstedt. Safe learning of quadrotor dynamics using barrier certificates. In *2018 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2460–2465. IEEE, 2018.

[45] E. R. Westervelt, J. W. Grizzle, C. Chevallereau, J. H. Choi, and B. Morris. *Feedback Control of Dynamic Bipedal Robot Locomotion*. Taylor & Francis/CRC Press, 2007.