Data-driven Iterative Optimal Control for Switched Dynamical Systems

Yuqing Chen, Yangzhi Li, and David J. Braun

Abstract— **This paper presents a data-driven algorithm to compute optimal control inputs for input constrained nonlinear optimal control problems with switched dynamics. We consider multi-stage optimal control problems where the control inputs and the switching instants are both unknown. Our key contribution is the new iterative online optimal control algorithm which mitigates sub-optimal control caused by model bias in the challenging class of under-actuated and intrinsically unstable switched dynamical systems. This is achieved by estimating the cost and computing the control inputs along measured trajectories of the controlled system instead of doing the same procedure along error-prone trajectories predicted by an inexact model. The algorithm is evaluated using an under-actuated and intrinsically unstable hopping robot in a simulation environment. The algorithm enables real-time data-driven optimal control using inaccurate models.**

I. Introduction

Switched systems are a class of hybrid dynamical systems that exhibit both continuous dynamics and discrete state transitions [1]. Robots that interact with the environment are often modeled as switched systems [2]. Models of these robots are composed of local continuous dynamics and discrete switching laws [3], [4]. Optimal control [5], [6] offers a systematic approach to control these robots via finding control inputs that minimize a user-defined cost.

There is an extensive literature on model-based optimal control of switched systems [7]. For control problems described by piecewise-affine dynamics and quadratic cost, optimal control leads to a piecewise-affine time-varying feedback control law [7], [8]. For a more general class of constrained nonlinear systems, the analytical form of the optimal solution is not known, and one typically resorts to efficient trajectory-based optimization to find locally optimal feedback control laws. For example, using the hybrid Maximum Principle [9], a two-stage iterative optimal control algorithm was proposed in [10] where the switching sequence is assumed to be known, while the optimal control inputs and the optimal switching instants are computed to reduce the cost. The two-stage approach has also been used to solve optimal switching time problems [11] and compute optimal controllers for robots with switching dynamics [12], [13]. In the alternative sequential action control (SAC) method [14], the authors proposed an optimal control formulation for trajectory tracking that admits analytical solution, and thereby an efficient real-time implementation. More recent works [15] extend SAC to high-dimensional robots with constrained inputs. It is common to the aforementioned model-based optimal control methods that (i) the cost is minimized using trajectories predicted by the model, or (ii) the cost is estimated along trajectories predicted by the model, where the former is done in model-predictive control (MPC) while the latter is done in model-based reinforcement learning (mRL).

While model-based methods can reduce the amount of measured data to compute the control inputs compared to model-free methods [16], modeling errors are ubiquitous, and inexact models lead to inaccurate future prediction and error-prone estimation of the future cost. The detrimental effect of modeling errors is evident in systems possessing continuous dynamics, but it is even more prominent in systems described by switched dynamics. This is because empirical switching laws typically poorly capture the physics of the complex system-environment interaction [17]. Mitigating the effect of model errors remains an important area of research, especially important in applications where model-free methods remain limited due to the lack of a large amount of measured data.

In this paper, we present a data-driven algorithm to solve optimal control problems defined by nonlinear cost, input constraints, and switched dynamics. The algorithm calculates the control inputs and estimates the cost without using error-prone predicted trajectories. The algorithm extends previously developed iterative optimal control methods [10], [12], [13], [18], [19] which use model-based future prediction to calculate the control input, and recently developed hardware-in-the-loop optimal control methods (HILOC) [20]–[22] which assume continuous dynamics.

Similar to model predictive control (MPC), model-based reinforcement learning (mRL) and iterative learning control (ILC) methods, the proposed method can be used for real-time robot control given an analytically derived, offline identified, or online learned inexact model. However, MPC methods approximate the optimal feedback controller by recomputing locally optimal feed-forward or feedback controllers starting from the measured state of the system [23]–[25];

Manuscript received: September 19, 2022; Accepted: November 12, 2022. Date of publication 1 December 2022; date of current version 8 December 2022. This letter was recommended for publication by Associate Editor K. N. Kaipa and Editor L. Pallottino upon evaluation of the reviewers' comments. The work of Yuqing Chen was supported by the Jiangsu Science and Technology Program under Grant BK20220283. This work was supported by the National Science Foundation CMMI DCSD under Grant 2029181. (*Corresponding author: D. J. Braun*)

Yuqing Chen is with the Xi'an Jiaotong-Liverpool University, Suzhou 215123, China (e-mail: yuqing.chen@xjtlu.edu.cn).

Yangzhi Li is with the Singapore University of Technology and Design, Singapore 487372 (e-mail: yangzhi_li@mymail.sutd.edu.sg).

David J. Braun is with the Advanced Robotics and Control Laboratory at the Department of Mechanical Engineering, Vanderbilt University, Nashville, TN 37235 USA (e-mail: david.braun@vanderbilt.edu).

This letter has supplementary downloadable material available at https://doi.org/10.1109/LRA.2022.3226075, provided by the authors.

Digital Object Identifier 10.1109/LRA.2022.3226075.

mRL methods learn a better model of the dynamics that leads to better state prediction and therefore a better controller [26]–[28]; while ILC methods use measured trajectory from previous iterations to calculate the control inputs for the next iteration [29]. Consequently, the main difference between MPC, mRL, and our proposed method is that the former two calculate the controller along an error-prone trajectory predicted by an estimated or learned inexact model [23]–[25], or estimates the cost using the trajectories predicted by an inexact model [26]–[28], while our proposed method calculates the controller and estimates the cost along the measured trajectory of the controlled system. ILC methods are by nature closer to our proposed method. However, ILC methods are developed to solve tracking problems [29], [30], which are optimal control problems that use offline computed and pre-defined reference trajectories as they do not involve online planning, while our method is developed to solve optimal control problems [22] which can involve online planning. In summary, this paper contributes to the algorithmic foundation of robotics by a novel algorithm suitable for real-time optimal control of robots characterized by switched dynamics.

The paper is structured as follows: In Section II, we introduce the mathematical formulation of the optimal control problem. In Section III, we describe the iterative nonlinear optimal control algorithm. In Section IV, we present the pseudocode of the algorithm. In Section V, we use the proposed algorithm to demonstrate stable locomotion of an under-actuated and open-loop unstable hopping robot. In Section VI, we discuss the benefits and limitations of the proposed method compared to prior works.

II. Problem formulation

We consider a switched dynamical system defined by K different local dynamics. We assume that the local dynamics switch according to a sequence

$$k \in \mathbb{K} \triangleq \{1, 2, \cdots, K\} \subset \mathbb{N}.$$

We assume that the switching among the local dynamics happens instantaneously at switching instants $T_k \in [0, T_K] \subset \mathbb{R}^+$. Between two adjacent switchings, the system evolves according to the continuous dynamics:

$$\dot{\mathbf{x}}(t) = \mathbf{f}_k(\mathbf{x}(t), \mathbf{u}(t)), \;\; t \in [T_{k-1}, T_k), \tag{1}$$

where $\mathbf{x}(t) \in \mathbb{R}^n$ is the state, $\mathbf{u}(t) \in \mathbb{R}^m$ is the control input. When switching happens, the state of the system changes according to the discrete switching law

$$\mathbf{x}^+(T_k) = \mathbf{g}_k(\mathbf{x}^-(T_k)) \in \mathbb{R}^n, \tag{2}$$

where the pre-switching and post-switching states are denoted by upper indexes $(*)^-$ and $(*)^+$, respectively. The switching can be time-based or state-based [10]. In the former case, the optimal switching times T_k are unconstrained whereas in the later case, they need to satisfy the following constraint:

$$\mathbf{h}_k(\mathbf{x}^-(T_k)) = \mathbf{0}. \tag{3}$$

In order to control the switched system (1)-(3), we formulate a multi-stage optimal control problem with free switching instants and constrained control inputs

$$\min_{\substack{\mathbf{u}(\cdot)\in\mathcal{U}\\ \mathbf{T}\in\mathbb{T}}} \sum_{k=1}^{K} \Big[\phi_k(\mathbf{x}^-(T_k)) + \int_{T_{k-1}}^{T_k} l_k(\mathbf{x}(t), \mathbf{u}(t))dt\Big], \tag{4}$$

$$\text{subject to } (1), (2), (3) \text{ and } \mathbf{x}(0) = \mathbf{x}_0, \tag{5}$$

where $\mathbf{u}(\cdot) : [0, T_K] \to \mathbb{R}^m$ is the control function, $\mathbf{T} = (T_1, T_2, ..., T_K)^\top \in \mathbb{R}^K$ is the vector of undetermined switching instants, $\phi_k : \mathbb{R}^n \to \mathbb{R}$ is the terminal cost at the k^{th} switching, $l_k : \mathbb{R}^n \times \mathbb{R}^m \to \mathbb{R}$ is the running cost between adjacent switching instants $[T_{k-1}, T_k)$, and $\mathbf{x}_0$ is the initial state. The admissible control function is bounded and measurable, and the time range associated with each local dynamics is bounded:

$$\mathcal{U} = \{\mathbf{u}(\cdot) : [0, T_K] \to \mathbb{R}^m | \; \mathbf{u}(\cdot) \text{ is measurable}, \quad \forall t \in [0, T_K] : \mathbf{u}(t) \in \mathbb{U} = [\mathbf{u}_{\min}, \mathbf{u}_{\max}]\}, \tag{6}$$

$$\mathbb{T} = \{\mathbf{T} \in \mathbb{R}^K \mid T^k_{\min} \le T_k - T_{k-1} \le T^k_{\max}\}, \tag{7}$$

where $\mathbf{u}_{\min} \prec \mathbf{0} \prec \mathbf{u}_{\max}$ and $\forall k \in \mathbb{K} : 0 < T^k_{\min} < T^k_{\max}$.

III. Optimal control with switching dynamics

We aim to iteratively solve the nonlinear optimal control problem given in (4)-(7), by solving a sequence of simpler constrained piecewise linear-quadratic sub-problems. In this section, we will derive the linear-quadratic sub-problem by (i) reducing (4)-(7) to a time-based switching problem (Section III-A), (ii) converting (4)-(7) to a fixed-switching-time problem (Section III-B), (iii) formulating the linear-quadratic approximation of (4)-(7) (Section III-C), and (iv) deriving the necessary conditions of local optimality for the sub-problem (Section III-D). The formulation enable us to introduce a data-driven iterative optimal control algorithm in Section IV.

A. Approximating State-based with Time-based Switching

One way to solve complex optimal control problems with state-based switching is to convert them into simpler optimal control problems with time-based switching. Such conversion can be done using the penalty method. The basic idea of the penalty method is to replace the switching constraint (3) with a switching cost. This can be done by augmenting the terminal cost in (4) with the state-based switching cost:

$$\phi^w_k(\mathbf{x}) = \phi_k(\mathbf{x}) + w_k \|\mathbf{h}_k(\mathbf{x})\|^2, \;\; w_k \in \mathbb{R}^+. \tag{8}$$

In this way, the original event-based switching problem (4)-(7) is approximated with a new time-based switching problem:

$$\min_{\substack{\mathbf{u}(\cdot)\in\mathcal{U}\\ \mathbf{T}\in\mathbb{T}}} \underbrace{\sum_{k=1}^{K} \Big[\phi^w_k(\mathbf{x}^-(T_k)) + \int_{T_{k-1}}^{T_k} l_k(\mathbf{x}(t), \mathbf{u}(t))dt\Big]}_{\mathcal{I}[\mathbf{u}(\cdot),\mathbf{T}]}, \tag{9}$$

$$\text{subject to } (1), (2) \text{ and } \mathbf{x}(0) = \mathbf{x}_0. \tag{10}$$

The last term in (8) makes (9)-(10) an inexact approximation of (4)-(7) where the approximation depends on the user-defined weights w_k. The following condition assure that the solution of (9)-(10) does not depend on w_k and is the exact solution of the original problem (4)-(7):

$$\forall w_k \in \mathbb{R}^+ : \|\mathbf{h}_k(\mathbf{x}^-(T_k))\|^2 = \mathbf{0}. \tag{11}$$

According to (11), if the switching happens while the state satisfies the switching constraint, then the optimal solution of the approximate time-based switching problem (9)-(10) will be the same as the optimal solution of the original state-based switching problem (4)-(7). Condition (11) can be used to numerically check if the solution obtained by (9)-(10) is the same as the optimal solution of the original problem (4)-(7).

B. Reducing a Free-Switching-Time to a Fixed-Switching-Time

In the next preparatory step, we reduce the free-switching time problem (9)-(10) to a fixed switching-time problem.

First, we convert the original time variable, t, into a new, normalized time variable:

$$\tau = (k-1) + \frac{t - T_{k-1}}{T_k - T_{k-1}} \in [0, K]. \tag{12}$$

Second, we introduce the augmented state $\mathbf{z}(\tau)$ and the new control input $\mathbf{u}(\tau)$:

$$\mathbf{z}(\tau) = \begin{bmatrix} \mathbf{x}(t(\tau)) \\ \mathbf{T} \end{bmatrix} \in \mathbb{R}^{n+K}, \;\; \mathbf{u}(\tau) \triangleq \mathbf{u}(t(\tau)) \in \mathbb{R}^m, \tag{13}$$

where $t(\tau) = T_{k-1} + (T_k - T_{k-1})(\tau - k + 1)$.

Finally, we transform the dynamics (1), the switching law (2), and the control costs (4) into:

$$\begin{aligned} \dot{\mathbf{z}}(\tau) &= \mathbf{F}_k(\mathbf{z}(\tau), \mathbf{u}(\tau)) \\ &= \begin{bmatrix} (T_k - T_{k-1})\mathbf{f}_k(\mathbf{x}(t(\tau)), \mathbf{u}(t(\tau))) \\ \mathbf{0} \end{bmatrix}, \; \tau \in [k-1, k), \end{aligned} \tag{14}$$

$$\mathbf{z}^+(k) = \mathbf{G}_k(\mathbf{z}^-(k)) = \begin{bmatrix} \mathbf{g}_k[\mathbf{x}^-(t(k))] \\ \mathbf{T} \end{bmatrix}, \tag{15}$$

where $\mathbf{z}(0) = [\mathbf{x}(0)^\top, \mathbf{T}^\top]^\top$, and

$$\Phi_k(\mathbf{z}^-(k)) = \phi_k^w(\mathbf{x}^-(t(k))), \tag{16}$$

$$L_k(\mathbf{z}(\tau), \mathbf{u}(\tau)) = (T_k - T_{k-1}) l_k(\mathbf{x}(t(\tau)), \mathbf{u}(t(\tau))). \tag{17}$$

In summary, the original formulation (9)–(10) has n states and K unknown switching times while the new formulation (14)–(17) has $n+K$ states and K unknown initial conditions.

C. Constrained Linear-Quadratic Subproblem

In order to derive the linear-quadratic sub-problem, we assume that a user-defined nominal control function $\mathbf{u}^i(\cdot) \in \mathcal{U}$ and switching instants $\mathbf{T}^i \in \mathbb{T}$ are given. Furthermore, we suppose that the control function is applied to the switched system, and that the resulting state trajectory is bounded

$$\mathbf{x}^i(\cdot) : [0, T_K] \to \mathcal{X}. \tag{18}$$

In the following we propose a data-driven optimal control approach where the state trajectory (18) is obtained by measuring the state of the controlled system instead of predicting the state by forward integration of the inexact system model.

As the first preparation step, we define the variation of the state $\delta\mathbf{x}(t) = \mathbf{x}(t) - \mathbf{x}^i(t)$, the switching instants $\delta\mathbf{T} = \mathbf{T} - \mathbf{T}^i$, the augmented state, and the control input:

$$\delta\mathbf{z}(\tau) = \mathbf{z}(\tau) - \mathbf{z}^i(\tau) = \left[\delta\mathbf{x}(t(\tau))^\top, \delta\mathbf{T}^\top\right]^\top, \tag{19}$$

$$\delta\mathbf{u}(\tau) = \mathbf{u}(\tau) - \mathbf{u}^i(\tau). \tag{20}$$

Subsequently, we derive the first- and second-order approximations of the dynamics (14)–(15) and the cost (16)–(17) around the nominal trajectory $\mathbf{z}^i(\cdot)$ as well as the nominal control function $\mathbf{u}^i(\cdot)$. The resulting constrained linear-quadratic sub-problem is given by:

$$\min_{\substack{\delta\mathbf{u}(\cdot)\in\delta\mathcal{U}^i \\ \delta\mathbf{T}\in\delta\mathbb{T}^i}} \underbrace{\sum_{k=1}^{K} \left[\Delta\Phi_k\big(k, \delta\mathbf{z}^-(k)\big) + \int_{k-1}^{k} \Delta L_k(\delta\mathbf{z}(\tau), \delta\mathbf{u}(\tau)) d\tau\right]}_{\Delta\mathcal{I}[\delta\mathbf{u}(\cdot), \delta\mathbf{T}]} \tag{21}$$

subject to:

$$\begin{aligned} &\delta\dot{\mathbf{z}}(\tau) = \mathbf{F}_{k,\mathbf{z}}(\tau)\delta\mathbf{z} + \mathbf{F}_{k,\mathbf{u}}(\tau)\delta\mathbf{u}, \; \tau \in [k-1, k), \\ &\delta\mathbf{z}^+(k) = \mathbf{G}_{k,\mathbf{z}}(k)\delta\mathbf{z}^-(k), \; \delta\mathbf{z}(0) = [\mathbf{0}^\top, \delta\mathbf{T}^\top]^\top, \end{aligned}$$

where

$$\Delta\Phi_k(\tau, \delta\mathbf{z}) = \Phi_{k,\mathbf{z}}(\tau)\delta\mathbf{z} + \frac{1}{2}\delta\mathbf{z}^\top \Phi_{k,\mathbf{zz}}(\tau)\delta\mathbf{z}, \tag{22}$$

$$\begin{aligned} \Delta L_k(\tau, \delta\mathbf{z}, \delta\mathbf{u}) = &\begin{bmatrix} L_{k,\mathbf{z}}(\tau) \\ L_{k,\mathbf{u}}(\tau) \end{bmatrix}^\top \begin{bmatrix} \delta\mathbf{z} \\ \delta\mathbf{u} \end{bmatrix} \\ &+ \frac{1}{2}\begin{bmatrix} \delta\mathbf{z} \\ \delta\mathbf{u} \end{bmatrix}^\top \begin{bmatrix} L_{k,\mathbf{zz}}(\tau) & L_{k,\mathbf{zu}}(\tau) \\ L_{k,\mathbf{uz}}(\tau) & L_{k,\mathbf{uu}}(\tau) \end{bmatrix} \begin{bmatrix} \delta\mathbf{z} \\ \delta\mathbf{u} \end{bmatrix}. \end{aligned} \tag{23}$$

In (21)–(23), the lower indexes $(*)_\mathbf{z}$, $(*)_\mathbf{u}$ denote partial derivatives with respect to $\mathbf{z}$ and $\mathbf{u}$. All partial derivatives are evaluated along the nominal state trajectory $\mathbf{z}^i(\cdot)$ and the nominal control function $\mathbf{u}^i(\cdot)$. The control constraints and the constraints imposed on the switching instants are given by:

$$\begin{aligned} \delta\mathcal{U}^i = \{&\delta\mathbf{u}(\cdot) : [0, T_K] \to \mathbb{R}^m | \; \delta\mathbf{u}(\cdot) \text{ is measurable}, \\ &\forall \tau \in [0, K] : \delta\mathbf{u}(\tau) \in \delta\mathbb{U}^i(\tau) = \mathbb{U} - \mathbf{u}^i(\tau)\}, \end{aligned} \tag{24}$$

$$\delta\mathbb{T}^i = \{\delta\mathbf{T} \in \mathbb{R}^K \mid T^k_{\min} - T^i_k \le \delta T_k \le T^k_{\max} - T^i_k\}. \tag{25}$$

D. Necessary Conditions of Optimality

We derive the necessary condition of optimality for the linear-quadratic sub-problem (21) using the Maximum Principle [6]. We first define the control Hamiltonian

$$\begin{aligned} \Delta H_k(\tau, \delta\mathbf{z}, \delta\mathbf{u}, \delta\boldsymbol{\lambda}) = &\Delta L_k(\tau, \delta\mathbf{z}, \delta\mathbf{u}) \\ &+ \delta\boldsymbol{\lambda}^\top\big(\mathbf{F}_{k,\mathbf{z}}(\tau)\delta\mathbf{z} + \mathbf{F}_{k,\mathbf{u}}(\tau)\delta\mathbf{u}\big), \end{aligned} \tag{26}$$

where $\delta\boldsymbol{\lambda} \in \mathbb{R}^{n+K}$ is the co-state. Subsequently, we derive the first-order optimality condition to find the control variation

$$\delta\mathbf{u}^i(\tau) = \underset{\delta\mathbf{u}\in\delta\mathbb{U}^i(\tau)}{\arg\min}\ \Delta H_k(\tau, \delta\mathbf{z}, \delta\mathbf{u}, \delta\boldsymbol{\lambda}), \tag{27}$$

where

$$\begin{aligned} \delta\dot{\boldsymbol{\lambda}} =& -\Delta H_{k,\delta\mathbf{z}}(\tau, \delta\mathbf{z}, \delta\mathbf{u}^i, \delta\boldsymbol{\lambda}), \\ \delta\boldsymbol{\lambda}^-(k) =& \Delta\Phi_{k,\delta\mathbf{z}}(k, \delta\mathbf{z}^-(k)) + \delta\boldsymbol{\lambda}^+(k). \end{aligned} \tag{28}$$

In order to convert (27) into a feedback control law, we approximate the co-state with a linear state-dependent time-varying function [18], [21]:

$$\delta\boldsymbol{\lambda} = \mathbf{P}(\tau)\delta\mathbf{z} + \mathbf{p}(\tau), \tag{29}$$

where $\mathbf{P}(\tau) \in \mathbb{R}^{(n+K)\times(n+K)}$ and $\mathbf{p}(\tau) \in \mathbb{R}^{n+K}$ are unknown time-dependent functions.

Substituting (29) into (27), we find that the time-dependent coefficients can be computed by the following piecewise-linear differential equations:

$$\begin{aligned} &\dot{\mathbf{P}} + \mathbf{P}\mathbf{F}_{k,\mathbf{z}} + \mathbf{F}_{k,\mathbf{z}}^\top\mathbf{P} + L_{k,\mathbf{zz}} = \mathbf{0},\ \mathbf{P}(K) = \Phi_{K,\mathbf{zz}}(K), \\ &\dot{\mathbf{p}} + (\mathbf{P}\mathbf{F}_{k,\mathbf{u}} + L_{k,\mathbf{zu}})\delta\mathbf{u}^{i-1} + \mathbf{F}_{k,\mathbf{z}}^\top\mathbf{p} = \mathbf{0},\ \mathbf{p}(K) = \Phi_{K,\mathbf{z}}(K), \\ &\mathbf{P}^-(k) = \Phi_{k,\mathbf{zz}}^-(k) + \mathbf{G}_{k,\mathbf{z}}^{\top-}(k)\mathbf{P}^+(k)\mathbf{G}_{k,\mathbf{z}}^-(k), \\ &\mathbf{p}^-(k) = \Phi_{k,\mathbf{z}}^-(k) + \mathbf{G}_{k,\mathbf{z}}^{\top-}(k)\mathbf{p}^+(k). \end{aligned} \tag{30}$$

The coefficients $\mathbf{P}(\tau)$ and $\mathbf{p}(\tau)$ are computed by backward integration of the first two equations in (30), starting from the known terminal conditions $\mathbf{P}(K) = \Phi_{K,\mathbf{zz}}(K)$, $\mathbf{p}(K) = \Phi_{K,\mathbf{z}}(K)$, and using the switching rules at $\tau = k \in \{K, K-1, ...2, 1, 0\}$, given by the last two equations in (30).

E. Updating $\mathbf{u}$ *and* $\mathbf{T}$

We compute the update of the switching instants by minimizing a second order expansion of the cost

$$\delta\mathbf{T}_\alpha^i = \underset{\delta\mathbf{T}\in\delta\mathbb{T}^i}{\arg\min}\ \underbrace{\frac{1}{2}\delta\mathbf{T}^\top\mathbf{P}_{22}(0)\delta\mathbf{T} + \alpha\mathbf{p}_2^\top(0)\delta\mathbf{T}}_{\delta\mathcal{I}_\alpha[\mathbf{0},\delta\mathbf{T}]} \tag{31}$$

where $\mathcal{I}[\mathbf{u}^i(\cdot), \mathbf{T}^{i+1}] - \mathcal{I}[\mathbf{u}^i(\cdot), \mathbf{T}^i] \approx \delta\mathcal{I}_{\alpha=1}[\mathbf{0}, \delta\mathbf{T}^i]$, while $\mathbf{P}_{11}(\tau) \in \mathbb{R}^{n\times n}$, $\mathbf{P}_{12}(\tau) \in \mathbb{R}^{n\times K}$, $\mathbf{P}_{22}(\tau) \in \mathbb{R}^{K\times K}$, $\mathbf{p}_1(\tau) \in \mathbb{R}^n$ and $\mathbf{p}_2(\tau) \in \mathbb{R}^K$ are defined by:

$$\mathbf{P}(\tau) = \begin{bmatrix} \mathbf{P}_{11}(\tau) & \mathbf{P}_{12}(\tau) \\ \mathbf{P}_{12}^\top(\tau) & \mathbf{P}_{22}(\tau) \end{bmatrix},\ \mathbf{p}(\tau) = \begin{bmatrix} \mathbf{p}_1(\tau) \\ \mathbf{p}_2(\tau) \end{bmatrix}. \tag{32}$$

Using (26)–(28), we compute the optimal control update by minimizing the approximate time, control, and state-dependent Hamiltonian

$$\delta\mathbf{u}_\alpha^i(\tau) = \underset{\delta\mathbf{u}\in\delta\mathbb{U}^i(\tau)}{\arg\min}\ \underbrace{\frac{1}{2}\delta\mathbf{u}^\top L_{k,\mathbf{uu}}\delta\mathbf{u} + \alpha\delta\mathbf{u}^\top\mathbf{b}(\tau, \delta\mathbf{x}^i, \delta\mathbf{T}_\alpha^i)}_{\Delta H_k^\alpha(\tau,\delta\mathbf{u},\delta\mathbf{z},\mathbf{P}(\tau)\delta\mathbf{z}+\mathbf{p}(\tau))} \tag{33}$$

where $\alpha \in (0, 1]$ is a convergence control parameter, $\delta\mathbf{x}(\tau) = \mathbf{x}^{i+1}(t(\tau)) - \mathbf{x}^i(t(\tau))$, $\mathbf{x}^{i+1}(\cdot)$ is the measured state while

$$\mathbf{b} = (L_{k,\mathbf{uz}} + \mathbf{F}_{k,\mathbf{u}}^\top\mathbf{P})\begin{bmatrix} \delta\mathbf{x}^i \\ \delta\mathbf{T}_\alpha^i \end{bmatrix} + L_{k,\mathbf{u}} + \mathbf{F}_{k,\mathbf{u}}^\top\mathbf{p}. \tag{34}$$

For a given $\alpha \in (0, 1]$, the new control function, and the new switching times are computed using

$$\mathbf{u}^{i+1} = \mathbf{u}^i + \delta\mathbf{u}_\alpha^i \quad \text{and} \quad \mathbf{T}^{i+1} = \mathbf{T}^i + \delta\mathbf{T}_\alpha^i. \tag{35}$$

The result is accepted if the cost is sufficiently decreased

$$\begin{aligned} \Delta\mathcal{I}[\delta\mathbf{u}^i(\cdot), \delta\mathbf{T}^i] &= \mathcal{I}[\mathbf{u}^{i+1}(\cdot), \mathbf{T}^{i+1}] - \mathcal{I}[\mathbf{u}^i(\cdot), \mathbf{T}^i] \\ &\le -c\left[\int_0^K \|\delta\mathbf{u}_\alpha^i(\tau)\|^2 d\tau + \|\delta\mathbf{T}_\alpha\|^2\right] \end{aligned} \tag{36}$$

where $c \in (0, \infty)$ is a positive constant.

The purpose of the convergence control parameter $\alpha \in (0, 1]$ in (31) and (33) is to ensure that the cost sufficiently decreases in every iteration, and therefore, it is similar to the line-search parameter in nonlinear programming [31].

If for $\alpha \in (0, 1]$ the inequality condition (36) is satisfied, then the cost is sufficiently decreased. When the cost sufficiently decreases, the control inputs and the switching instants are improved and the iteration is deemed successful. If sufficient decrease of the cost (36) is not achieved, then the convergence control parameter α is reduced using backtracking [31] until the cost is sufficiently decreased. Otherwise, the computation is terminated with the control law and switching instants from the previous iteration.

IV. Online Optimal Control Algorithm

In this section, we consolidate the formulas presented in Section III-E into an iterative optimal control algorithm summarized in Algorithm 1.

In Algorithm 1 we make the following two assumptions:

(i) The state trajectory (18) is measured.

(ii) The exact model $\mathbf{F}_k$ and $\mathbf{G}_k$ in (30) and (34) is not known. Consequently, it is replaced with an inexact model $\hat{\mathbf{F}}_k$ and $\hat{\mathbf{G}}_k$ defined by (14) and (15) using:

$$\forall k \in \mathbb{K}:\ \ \dot{\mathbf{x}} = \hat{\mathbf{f}}_k(\mathbf{x}, \mathbf{u}),\ \text{ and }\ \mathbf{x}^+ = \hat{\mathbf{g}}_k(\mathbf{x}^-). \tag{37}$$

This models can be derived by first principles, estimated via offline system identification, or learned from data online.

Algorithm 1 is divided into (i) offline computation (red) performed between iterations, and (ii) online computation (blue) performed during each iteration. The offline computation involves backward integration of $\frac{1}{2}n(n-1)+n$ piecewise linear differential equations (30) to find the co-state used in the subsequent online iteration (29). The switching times are also computed offline between iterations (31). During the online computation, the Hamiltonian is minimized under the control constraints – we solve a box-constrained quadratic program (33) at each time step. The online computational cost scales cubically with the number of control inputs $\mathcal{O}(m^3)/time\_step$ but is independent of the time horizon and the number of states. This feature makes Algorithm 1 feasible for system controlled with large number of inputs and small sampling time Δt (for example, $m = 20$ and $\Delta t = 0.001$ s, see [21]).

Algorithm 1: Data-Driven Iterative Optimal Control of Switched Dynamical Systems

Input: $\mathbf{u}^0(\cdot) \in \mathcal{U}$, $\mathbf{T}^0 \in \mathbb{T}$, $\hat{\mathbf{f}}_k(\mathbf{x},\mathbf{u})$, $\hat{\mathbf{g}}_k(\mathbf{x})$
Output: Near-optimal solution $\mathbf{u}^{\text{cnvg}}(\cdot) \in \mathcal{U}$, $\mathbf{T}^{\text{cnvg}} \in \mathbb{T}$
Initialize: $cnvg = false$, $i = 0$, $\gamma \in (0,1)$, $c \in (0,\infty)$, $0 < \epsilon \ll 1$, $\mathbf{x}^0(\cdot) \leftarrow \big(\text{system}, \mathbf{u}^0(\cdot), \mathbf{T}^0\big)$, $\mathcal{I}^0 = \mathcal{I}[\mathbf{u}^0(\cdot), \mathbf{T}^0]$, $\delta\mathbf{u}^0(\cdot) = \mathbf{0}$, $\Delta i \in \mathbb{N}$
while $cnvg = false$ **do**
  Offline computation (between iterations):
  for $j = n_T : -1 : 2$ **do**
    Evaluate: $\hat{\mathbf{F}}_{k,\mathbf{z}}$, $\hat{\mathbf{F}}_{k,\mathbf{u}}$, $L_{k,\mathbf{z}}$, $L_{k,\mathbf{u}}$, $L_{k,\mathbf{zz}}$, $L_{k,\mathbf{uu}}$, $L_{k,\mathbf{uz}}$ using $\mathbf{z}_j^i$, $\mathbf{u}_j^i$
    Integrate: $\mathbf{P}_{j-1}^i$, $\mathbf{p}_{j-1}^i$
  end
  Set $\alpha^i = 1$, $iter = true$
  Online implementation (during iterations):
  while $iter = true$ **do**
    Update: Switching instant $\delta\mathbf{T}_\alpha^i \leftarrow (31), \alpha^i$
    $\mathbf{T}^{i+1} = \mathbf{T}^i + \delta\mathbf{T}_\alpha^i$
    for $j = 1 : n_T$ **do**
      Measure: $\mathbf{x}_j^{i+1} \leftarrow$ system; $\delta\mathbf{x}_j^i = \mathbf{x}_j^{i+1} - \mathbf{x}_j^i$
      Compute: $\delta\mathbf{u}_{\alpha,j}^i \leftarrow (33)$, $\delta\mathbf{x}_j^i$, $\delta\mathbf{T}_\alpha^i$, α^i
      Control: $\mathbf{u}_j^{i+1} = \mathbf{u}_j^i + \delta\mathbf{u}_{\alpha,j}^i$; apply $\mathbf{u}_j^{i+1}$ to the system
    end
    Measure: $\mathbf{T}^{i+1}$
    Compute: $\delta\mathbf{T}_\alpha^i = \mathbf{T}^{i+1} - \mathbf{T}^i$
    $\mathcal{I}^{i+1}, \Delta\mathcal{I}^i \leftarrow (36), \mathbf{x}^{i+1}(\cdot), \mathbf{u}^{i+1}(\cdot), \mathbf{T}^{i+1}$
    if $\Delta\mathcal{I}^i \le -c[\int_0^K \|\delta\mathbf{u}_\alpha^i(\tau)\|^2 d\tau + \|\delta\mathbf{T}_\alpha^i\|^2]$ **then**
      $iter = false$
    else
      $\alpha^i = \gamma\alpha^i$
    end
  end
  if $\forall i \in \{i^*, i^* + \Delta i\}$: $\int_0^K \|\delta\mathbf{u}_\alpha^i(\tau)\|^2 d\tau + \|\delta\mathbf{T}_\alpha^i\|^2 \le \epsilon$ **then**
    $\mathbf{u}^{\text{cnvg}} = \mathbf{u}^{i+1}$, $\mathbf{T}^{\text{cnvg}} = \mathbf{T}^{i+1}$, $cnvg = true$
  end
  $i = i + 1$
end

V. Learning Locomotion

In this section, we use Algorithm 1 to iteratively learn hopping locomotion driven by two variable stiffness actuators (VSA) [19] (Fig. 1). The one legged hopper [32], [33] is a redundantly under-actuated and statically unstable system that exhibits nonlinear hybrid dynamics. We assume that the best model of the robot (one that cannot be further improved by model learning) is inexact just as it would be in practical application. Under this assumption, we aim to show that Algorithm 1 can iteratively learn stable hopping locomotion while being robust enough to moderate external disturbances. In what follows, we define the model of the robot (Section V-A), the optimal control cost to represent locomotion (Section V-B) and the learned motion (Section V-D). The results are compared to the result obtained using optimal feedback control [12] and iterative model-based reinforcement learning [26] applied to the same problem.

A. Model

The planar, one-legged hopping robot is shown in Fig. 1. The robot consists of a body of mass M and moment of inertia J_{body}, and a leg which has negligible mass compared to the body [34], but non-negligible moment of inertia J_{leg} [33].

The hopper is actuated with two variable stiffness actuators; one rotating the hip and one extending the leg (Fig. 1). The motion of the hopper is described by the Cartesian coordinates of the center-of-mass (x, y), the rotation angle of the body ϕ, the rotation angle of the leg θ, and the length of the leg r. The model parameters are given in Table I.

TABLE I

Parameter	Symbol	Value	Unit
Mass	M	{8,12}	kg
Body inertia	J_{body}	2.5	kg·m^2
Leg inertia	J_{leg}	0.25	kg·m^2
Leg length	r_0	0.7	m
Hip torque range	τ_{hip}	[-4,4]	N·m
Hip stiffness range	k_{hip}	[10,150]	N·m
Leg force range	F_{leg}	[-4,4]	N
Leg stiffness range	k_{leg}	[1780,2350]	N·m^{-1}
Air resistance	c_d	0.01	kg·m^{-1}

The state-space representation of the model is given by $\dot{\mathbf{x}} = \mathbf{f}_{k\in\{1,2\}}(\mathbf{x},\mathbf{u})$ and $\mathbf{x}^+ = \mathbf{g}_{k\in\{1,2\}}(\mathbf{x}^-)$ where the dynamics switch between *flight phase* $k = 1 : \mathbf{x} = [x, y, \theta, \phi, r, k_1, k_2, \dot{x}, \dot{y}, \dot{\theta}, \dot{\phi}, \dot{r}, \dot{k}_1, \dot{k}_2]^\top \in \mathbb{R}^{14}$, and *stance phase* $k = 2 : \mathbf{x} = [\theta, \phi, r, k_1, k_2, \dot{\theta}, \dot{\phi}, \dot{r}, \dot{k}_1, \dot{k}_2]^\top \in \mathbb{R}^{10}$. The control input is: $\mathbf{u} = [\tau_{0\text{hip}}, F_{0\text{leg}}, k_{\text{hip}}, k_{\text{leg}}]^\top \in \mathbb{R}^4$.

Figure 1 shows the flight and stance phase motions, and the discrete switching at touch-down and take-off. The flight phase dynamics is defined by

$$\begin{aligned} &\ddot{x} = -\frac{c_d}{M}(\dot{x}|\dot{x}|),\ \ddot{y} = -g - \frac{c_d}{M}(\dot{y}|\dot{y}|), \\ &\ddot{\theta} = -\tau_{\text{hip}}/J_{\text{leg}},\ \ddot{\phi} = \tau_{\text{hip}}/J_{\text{body}},\ \ddot{r} = 0, \end{aligned} \tag{38}$$

where g denotes the gravitational acceleration, τ_{hip} is the control torque applied at the hip joint, while c_d is the air drag coefficient [35]. The stance phase dynamics is governed by

$$\begin{aligned} &\ddot{\theta} = [M(rg\sin\theta - 2r\dot{r}\dot{\theta}) - \tau_{\text{hip}}]/(J_{\text{leg}} + Mr^2), \\ &\ddot{\phi} = \tau_{\text{hip}}/J_{\text{body}},\ \ddot{r} = (r\dot{\theta}^2 - g\cos\theta) + F_{\text{leg}}/M, \end{aligned} \tag{39}$$

where r_0 is the natural length of the leg while F_{leg} is the control force used to change the length of the leg. We assume that the hip torque τ_{hip}, the leg force F_{leg}, the hip stiffness

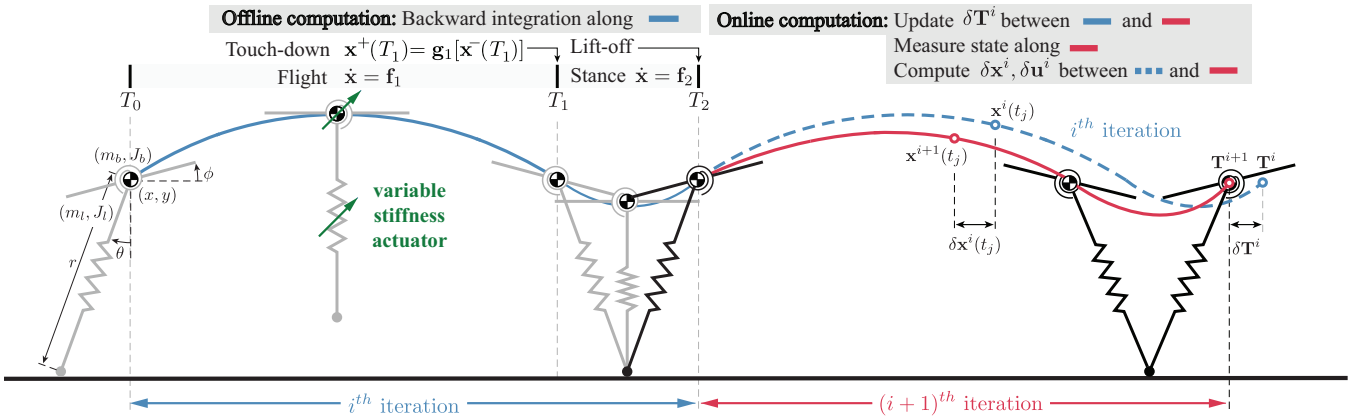

Fig. 1. Hopping locomotion implemented with Algorithm 1. During the iterative optimization, the trajectory in the previous step is used to compute the controller and evaluate the reduction of the cost for the next step.

k_{hip} and the leg stiffness k_{leg} are provided by two variable stiffness actuators (Fig. 1) [19]:

$$\tau_{\text{hip}} = \tau_{0\text{hip}} - k_{\text{hip}}(\phi - \theta), \quad \ddot{k}_{\text{hip}} + 2\beta\dot{k}_{\text{hip}} + \beta^2 k_{\text{hip}} = \beta^2 k_{0\text{hip}},$$
$$F_{\text{leg}} = F_{0\text{leg}} - k_{\text{leg}}(r - r_0), \quad \ddot{k}_{\text{leg}} + 2\beta\dot{k}_{\text{leg}} + \beta^2 k_{\text{leg}} = \beta^2 k_{0\text{leg}},$$

where $\tau_{0\text{hip}}$ and $F_{0\text{leg}}$ are the desired control torque and force, $k_{0\text{hip}}$ and $k_{0\text{leg}}$ are the desired hip and leg stiffness, and $\beta \in (0, \infty)$ is a parameter that defines how fast can the actuators modulate stiffness.

The switching between the flight-phase dynamics (38) and stance-phase dynamics (39) occurs when the leg touches the ground at T_1 or leaves the ground at T_2. At T_2, the dynamics switch from stance to flight phase by breaking the foot-ground contact without impact. At T_1, the dynamics switch from flight to stance phase through a single-contact impact. We assume that the impact is plastic [36], meaning the pre-impact velocities $(*)^-$ instantaneously change into post-impact velocities $(*)^+$ according to:

$$\begin{bmatrix} \dot{x}^+ \\ \dot{y}^+ \\ \dot{\theta}^+ \end{bmatrix} = \begin{bmatrix} \dot{x}^- \\ \dot{y}^- \\ \dot{\theta}^- \end{bmatrix} - \frac{J_{\text{body}}}{J_{\text{body}} + Mr_0^2} \begin{bmatrix} \cos\theta \\ \sin\theta \\ \frac{Mr_0}{J_{\text{body}}} \end{bmatrix} v_{\text{f}\theta}^-, \tag{40}$$

where $v_{\text{f}\theta}^- = \dot{x}^- \cos\theta + \dot{y}^- \sin\theta + r_0\dot{\theta}^-$ is the pre-impact velocity of the foot perpendicular to the leg. Due to the non-negligible rotational inertia of the leg, the kinetic energy loss at touchdown is [33], [37]:

$$\Delta E = |E^+ - E^-| = \frac{1}{2} M \frac{J_{\text{body}}}{J_{\text{body}} + Mr_0^2} (v_{\text{f}\theta}^-)^2. \tag{41}$$

We aim to minimize this loss to achieve efficient locomotion.

B. Task

We use the following cost to model hopping locomotion:

$$\mathcal{I} = \underbrace{\big(\mathbf{x}(T_2) - \mathbf{x}(0)\big)^\top \mathbf{Q} \big(\mathbf{x}(T_2) - \mathbf{x}(0)\big)}_{\mathcal{I}_1} + \underbrace{\frac{1}{2}\int_0^{T_2} \mathbf{u}(t)^\top \mathbf{R}\mathbf{u}(t)dt}_{\mathcal{I}_2} + \underbrace{\Delta E}_{\mathcal{I}_3} + \underbrace{w y_{\text{f}}(T_1)^2}_{\mathcal{I}_4}, \tag{42}$$

where $\mathcal{I}_1$ encourages periodicity by penalizing the discrepancies between the initial conditions through subsequent steps, $\mathcal{I}_2$ reduces the magnitude of the control inputs to encourage energetically passive hopping, $\mathcal{I}_3$ encourages efficient locomotion by penalizing collision energy loss at touchdown (41) while $\mathcal{I}_4$ encourages event driven switching by penalizing non-zero vertical distance between the foot and the ground y_{f}. The last term represents the penalty term in (8) and (9).

C. Online Learning

Figure 1 shows the implementation of Algorithm 1. In this implementation, the cost defined in (42) is the cost of a single step, and Algorithm 1 uses the measured trajectory from the previous step (dashed blue line) to calculate the unknown parameters (30) in the quadratic program (31)-(33), and to compute the feedback control inputs in the current step (red line). Consequently, in our method, the measured trajectory in the previous step (blue line) is used to calculate the control inputs for the next step (red line). This approach does not require trajectory prediction using the inexact model of the system to estimate the cost and to calculate the control inputs.

D. Results

The results are shown in Fig. 2 and the supplementary video. Below we summarized the results.

Exact model: We used Algorithm 1 with two different masses $M = 8$ kg and $M = 12$ kg, nonzero air resistance $c_d = 0.01$, and the parameters given in Table I. Algorithm 1 was able to learn stable locomotion. Algorithm 1 required an initial control sequence that achieves stable locomotion for the iterative learning to converge. The results are not presented.

Inexact model: Subsequently, we used Algorithm 1 with mass $\hat{M} = 10$ kg and no air resistance $\hat{c}_d = 0$ when computing the control inputs for robots of mass $M = 8$ kg and $M = 12$ kg subject to air resistance of $c_d = 0.01$. The results are shown in Fig. 2 (blue and red lines). We observed that, despite the deliberately introduced model errors, Algorithm 1 successfully learned stable locomotion.

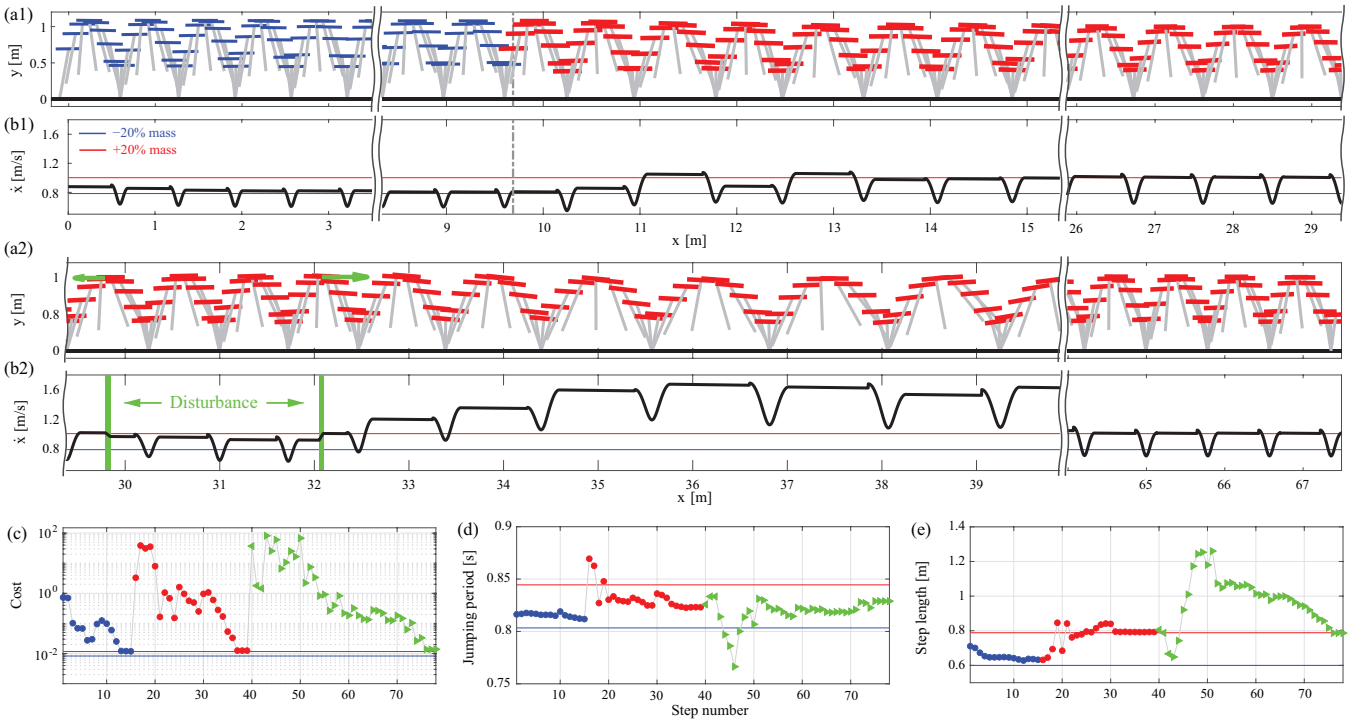

Fig. 2. Hopping locomotion controlled with Algorithm 1. (a) Jumping motion. (b) Speed $\dot{x}$. (c) Cost. (d) Jumping period T_2. (e) Step length. We use $\mathbf{u}_{\min} = [-4, -4, 10, 1780]$ and $\mathbf{u}_{\max} = [4, 4, 150, 2350]$. The initial control $\mathbf{u}_0$ is defined using the model-based optimal control function. During the motion, the robot was perturbed with a backward push $F_b = -20$ N and a forward push $F_f = 25$ N. Weights for cost in (42) are defined by $\mathbf{Q} = \text{diag}([0, 0, 600, 1000, 0, 0, 0, 1000, 10, 1, 1, 0, 0, 0])$, $\mathbf{R} = 10^{-2} \times \text{diag}([1, 10^{-2}, 10^{-4}, 10^{-4}])$, $w = 10^{-3}$. The motion can be seen in the supplementary video.

Disturbance rejection: We applied a horizontal force $F_b = -20$ N to the center of mass to push the body backwards and a subsequent horizontal force $F_f = 25$ N to push the body forward (Fig. 2). In these simulations, Algorithm 1 was used with the inexact model $\hat{M} = 10$ kg and assumed no air resistance $\hat{c}_d = 0$, while the robot was $+20\%$ heavier $M = 12$ kg and was subject to air resistance $c_d = 0.01$. The results are shown in Fig. 2 (green lines). Despite the combined effect of the inexact model and the external disturbances, Algorithm 1 provided stable hopping locomotion.

Optimality: We have previously noted that the cost (42) contains a penalty term $\mathcal{I}_4$ that encourages the optimal time-based switching to coincide with event-based switching at touchdown and takeoff. This penalty term increases the cost and leads to sub-optimality in problems where the optimal time-based switching differs from the optimal state-based switching. However, in our example, Algorithm 1 led to zero penalty at convergence, $\mathcal{I}_4 = 0$, implying that the optimal time-based switching coincides with the optimal state-based switching in the considered example.

Benchmark: We used the method presented in [12] to compute the model-based optimal feedback controller (OC-FB). In this benchmark, we used the same cost and the same model as in the aforementioned simulations. We found that *when the model was exact*, the OC-FB method was able to generate stable locomotion, and the motion was identical to the one obtained using Algorithm 1. However, *when the model was inexact*, as in Fig. 2, the OC-FB method was unable to generate stable locomotion, even when we discarded external disturbances; please see the supplementary video. This result shows the benefit of using optimal control without model-based future prediction [21].

Comparison: Subsequently, we compared Algorithm 1 with the inexact model-based reinforcement learning mRL algorithm proposed in [26]. In both algorithms we used the same inexact model; we assume that the model is the best inexact model that can be learned from data. We make this assumption to isolate the difference between the two methods that comes solely from the difference in the optimal control calculation, and therefore unambiguously demonstrate the advantage of eliminating model-based future prediction.

In our implementation, the mRL algorithm generated identical results to Algorithm 1 when the model was exact. When the model was inexact, as in Fig. 2, the mRL algorithm could generate a couple of steps but was unable to converge to stable hopping locomotion; please see the supplementary video. The mRL algorithm performed better than the OC-FB method because it used line-search based on measured trajectories to refine the optimal control calculation, and therefore decrease the cost in every iteration. However, the mRL calculated the search direction based on the estimated cost where the estimation was done along model-based predicted trajectories. A search direction computed using inexact model based predicted trajectories can significantly differ

from the optimal search direction computed in Algorithm 1 along the measured system trajectories. The difference between mRL and our proposed method demonstrate the benefit of using measured trajectories as opposed to inexact-model based predicted trajectories, to calculate both, the control inputs and the search direction [22].

VI. Conclusion

The main contribution of this paper is the data-driven iterative optimal control algorithm applicable to switched dynamical systems (Section IV). The key novelty of this algorithm is that it can be used to compute the control inputs and the switching instants using measured trajectories of the controlled system instead of using error-prone model predicted trajectories obtained by forward integration of an inexact model. We conjectured that Algorithm 1 can reduce the cost and improve the robustness of the controlled system against model uncertainty and external disturbances, especially compared to optimal control methods that use model-based future prediction. Our conjecture is supported by the simulation in Section V, where we use Algorithm 1 to learn stable hopping locomotion.

An alternative to the proposed method are model-free reinforcement learning (RL) methods that learn the model of the value function [38]–[40] instead of learning the model of the dynamics. These RL methods are conceptually similar to our method, partly because they do not use model-based future prediction, but also because the sub-optimality in both of these methods is due to modeling error. In particular, the sub-optimality in the aforementioned RL methods is due to the error in the model of the value function, as the value function cannot be exactly learned for constrained nonlinear and finite time horizon optimal control problems considered in this paper. On the other hand, the sub-optimality in our method is due to the error in the dynamics, because the dynamics cannot be exactly learned. Learning a value function that can furnish a near-optimal feedback controller requires complex neural network function approximators and large amounts of training data, which make value-function-based optimal control methods well suited to low dimensional and unconstrained optimal control problems, as in these problems exact parametrization of the value function can be done using finite-dimensional function approximation [40]. The same is not feasible for the optimal control problem considered in this paper.

References

[1] B. Brogliato, *Nonsmooth Mechanics*. Springer International Publishing, 2016.

[2] A. M. Johnson, S. A. Burden, and D. E. Koditschek, "A hybrid systems model for simple manipulation and self-manipulation systems," *The International Journal of Robotics Research*, vol. 35, no. 11, pp. 1354–1392, 2016.

[3] C. D. Remy, "Ambiguous collision outcomes and sliding with infinite friction in models of legged systems," *The International Journal of Robotics Research*, vol. 36, no. 12, pp. 1252–1267, 2017.

[4] Y. Li, H. Yu, and D. J. Braun, "Algorithmic resolution of multiple impacts in nonsmooth mechanical systems with switching constraints," in *IEEE International Conference on Robotics and Automation (ICRA)*, (Montreal, Canada), pp. 7639–7645, May 2019.

[5] R. Bellman, *Dynamic Programming*. Princeton, NJ, USA: Princeton University Press, 1 ed., 1957.

[6] L. Pontryagin, V. Boltyanskii, R. Gamkrelidze, and E. Mishchenko, *The Mathematical Theory of Optimal Processes*. John Wiley and Sons Inc., 1962.

[7] F. Zhu and P. J. Antsaklis, "Optimal control of hybrid switched systems: A brief survey," *Discrete Event Dynamic Systems*, vol. 25, no. 3, pp. 345–364, 2015.

[8] M. Barić, P. Grieder, M. Baotić, and M. Morari, "An efficient algorithm for optimal control of PWA systems with polyhedral performance indices," *Automatica*, vol. 44, no. 1, pp. 296–301, 2008.

[9] A. E. Bryson and Y.-C. Ho, *Applied Optimal Control: optimization, estimation and control*. CRC Press, 1975.

[10] X. Xu and P. J. Antsaklis, "Optimal control of switched systems based on parameterization of the switching instants," *IEEE Transactions on Automatic Control*, vol. 49, no. 1, pp. 2–16, 2004.

[11] H. Gonzalez, R. Vasudevan, M. Kamgarpour, S. S. Sastry, R. Bajcsy, and C. Tomlin, "A numerical method for the optimal control of switched systems," in *IEEE Conference on Decision and Control (CDC)*, pp. 7519–7526, 2010.

[12] J. Nakanishi, A. Radulescu, D. J. Braun, and S. Vijayakumar, "Spatio-temporal stiffness optimization with switching dynamics," *Autonomous Robots*, vol. 41, no. 2, pp. 273–291, 2017.

[13] N. J. Kong, G. Council, and A. M. Johnson, "iLQR for piecewise-smooth hybrid dynamical systems," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 5374–5381, 2021.

[14] A. R. Ansari and T. D. Murphey, "Sequential action control: Closed-form optimal control for nonlinear and nonsmooth systems.," *IEEE Trans. Robotics*, vol. 32, no. 5, pp. 1196–1214, 2016.

[15] E. Tzorakoleftherakis and T. D. Murphey, "Iterative sequential action control for stable, model-based control of nonlinear systems," *IEEE Transactions on Automatic Control*, vol. 64, no. 8, pp. 3170–3183, 2019.

[16] J. Kober, J. A. Bagnell, and J. Peters, "Reinforcement learning in robotics: A survey," *The International Journal of Robotics Research*, vol. 32, no. 11, pp. 1238–1274, 2013.

[17] B. Brogliato, A. ten Dam, L. Paoli, F. Génot, and M. Abadie, "Numerical simulation of finite dimensional multibody nonsmooth mechanical systems," *Applied Mechanics Reviews*, vol. 55, no. 2, pp. 107–150, 2002.

[18] W. Li and E. Todorov, "Iterative linearization methods for approximately optimal control and estimation of non-linear stochastic system," *International Journal of Control*, vol. 80, no. 9, pp. 1439–1453, 2007.

[19] D. J. Braun, F. Petit, F. Huber, S. Haddadin, P. van der Smagt, A. Albu-Schaffer, and S. Vijayakumar, "Robots driven by compliant actuators: Optimal control under actuation constraints," *IEEE Transactions on Robotics*, vol. 29, no. 5, pp. 1085–1101, 2013.

[20] Y. Chen, L. Roveda, and D. J. Braun, "Efficiently computable constrained optimal feedback controllers," *IEEE Robotics and Automation Letters*, vol. 4, no. 1, pp. 121–128, 2019.

[21] Y. Chen and D. J. Braun, "Hardware-in-the-loop iterative optimal feedback control without model-based future prediction," *IEEE Transactions on Robotics*, vol. 35, no. 6, pp. 1419–1434, 2019.

[22] Y. Chen and D. J. Braun, "Iterative online optimal feedback control," *IEEE Transactions on Automatic Control (Accepted)*, 2020.

[23] E. F. Camacho, D. R. Ramírez, D. Limón, D. M. De La Peña, and T. Alamo, "Model predictive control techniques for hybrid systems," *Annual Reviews in Control*, vol. 34, no. 1, pp. 21–31, 2010.

[24] F. Borrelli, A. Bemporad, and M. Morari, *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[25] U. Rosolia and A. D. Ames, "Iterative model predictive control for piecewise systems," *IEEE Control Systems Letters*, vol. 6, pp. 842–847, 2022.

[26] P. Abbeel, M. Quigley, and A. Y. Ng, "Using inaccurate models in reinforcement learning," in *International Conference on Machine Learning*, (Pittsburgh, Pennsylvania, USA), pp. 1–8, 2006.

[27] S. Levine and V. Koltun, "Guided policy search," in *International Conference on Machine Learning*, (Atlanta, USA), pp. 1–9, 2013.

[28] B. Thananjeyan, A. Balakrishna, U. Rosolia, F. Li, R. McAllister, J. E. Gonzalez, S. Levine, F. Borrelli, and K. Goldberg, "Safety augmented value estimation from demonstrations (SAVED): Safe deep model-based rl for sparse cost robotic tasks," *IEEE Robotics and Automation Letters*, vol. 5, no. 2, pp. 3612–3619, 2020.

[29] D. A. Bristow, M. Tharayil, and A. G. Alleyne, "A survey of iterative

learning control," *IEEE Control Systems*, vol. 26, no. 3, pp. 96–114, 2006.
[30] N. Amann, D. H. Owens, and E. Rogers, "Iterative learning control using optimal feedback and feedforward actions," *International Journal of Control*, vol. 65, no. 2, pp. 277–293, 1996.
[31] J. Nocedal and S. Wright, *Numerical Optimization*. Springer Science & Business Media, 2006.
[32] C. François and C. Samson, "A new approach to the control of the planar one-legged hopper," *The International Journal of Robotics Research*, vol. 17, pp. 1150–1166, nov 1998.
[33] S.-H. Hyon and T. Emura, "Energy-preserving control of a passive one-legged running robot," *Advanced Robotics*, vol. 18, no. 4, pp. 357–381, 2004.
[34] M. Garcia, "The simplest walking model: Stability, complexity, and scaling," *Journal of Biomechanical Engineering*, vol. 120, pp. 281–288, apr 1998.
[35] A. V. Hill, "The air-resistance to a runner," *Proceedings of the Royal Society B: Biological Sciences*, vol. 102, pp. 380–385, feb 1928.
[36] C. Glocker, "An introduction to impacts," in *Nonsmooth Mechanics of Solids*, pp. 45–101, Springer Vienna, 2006.
[37] F. Pfeiffer and C. Glocker, *Multibody dynamics with unilateral contacts*, vol. 9. John Wiley & Sons, 1996.
[38] A. Heydari and S. N. Balakrishnan, "Optimal switching and control of nonlinear switching systems using approximate dynamic programming," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 6, pp. 1106–1117, 2013.
[39] H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, and V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost," in *IEEE International Conference on Robotics and Automation*, pp. 3651–3657, May 2019.
[40] B. Recht, "A tour of reinforcement learning: The view from continuous control," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 2, no. 1, pp. 253–279, 2019.