

# Reference-Steering via Data-Driven Predictive Control for Hyper-Accurate Robotic Flying-Hopping Locomotion

Yicheng Zeng\*, Yuhao Huang\*, and Xiaobin Xiong

**Abstract**—State-of-the-art model-based control designs have been shown to be successful in realizing dynamic locomotion behaviors for robotic systems. The precision of the realized behaviors in terms of locomotion performance via fly, hopping, or walking has not yet been well investigated, despite the fact that the difference between the robot model and physical hardware is doomed to produce inaccurate trajectory tracking. To address this inaccuracy, we propose a referencing-steering method to bridge the *model-to-real gap* by establishing a data-driven input-output (DD-IO) model on top of the existing model-based design. The DD-IO model takes the *reference tracking trajectories as the input* and the *realized tracking trajectory as the output*. By utilizing data-driven predictive control, we steer the reference input trajectories online so that the realized output ones match the actual desired ones. We demonstrate our method on the robot PogoX to realize hyper-accurate hopping and flying behaviors in both simulation and hardware. This data-driven reference-steering approach is straightforward to apply to general robotic systems for performance improvement via hyper-accurate trajectory tracking.

## I. INTRODUCTION

Model-based control design has been the core approach to realizing trustworthy dynamic, efficient, and safe behaviors on modern robotic systems, especially on flying [1], [2] and legged robots [3]. The robot model being used, despite being either simplified or comprehensive such as the full-order Lagrangian dynamics model, describes the essential dynamics of how the input force/torques influence the states of the robot. Model-based controllers such as state feedback controllers [4]–[7] and Model Predictive Control [8]–[10] utilize the robot model to optimally plan and stabilize the trajectories of the robot for realizing locomotion tasks.

The models derived from theoretical physics laws are expected to capture the actual dynamics of the robot to some good extent but never exactly, since the actual robot can have complex components and processes that may not be feasible to model, producing this *model-to-real gap*. For instance, the robot linkages can have inaccuracy physical properties and they deform under load, the lower-level motor controls may not be fast enough to realize the desired torque or motor speed, and the sensing and computation loops have delays [11], [12]. Additionally, in real-world deployment, there will also be uncertainties on the hardware from the environment that we cannot model. Therefore, it is expected that in the model-based control design process, the realized

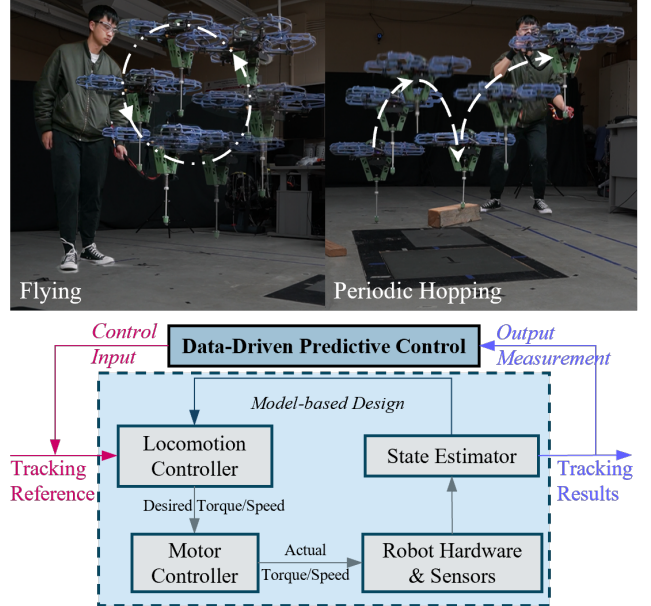


Fig. 1: Overview of the data-driven reference-steering approach (bottom) and the application on PogoX (top).

trajectories can be different than the reference trajectories, yet within a reasonable range of tracking errors.

Data-driven and learning-based approaches [13]–[23] have been the alternatives to address the problems of lack accurate models. Roughly speaking, they are concerned with data-driven models of the systems [13], [15], [17], [24], controllers [16], [21], or combinations of them [18], [25]. By and large, the data-driven and learning-based approaches either learn the controller or require re-synthesizing the control design based on the learned model, both of which abandon the original model-based control design that has proven theoretical properties.

In this work, we aim to utilize the data-driven predictive control (DDPC) techniques but keep the original model-based control design intact to address tracking errors caused by the model-to-real gap in robotic locomotion control. Fig. 1 illustrates our framework. We assume a model-based control design, presented by the dashed box, has been synthesized to realize robotic locomotion by performing trajectory tracking. The model-to-real gap causes errors between the reference and resultant trajectories. We then take a perspective of treating the reference trajectory as *the input* and the resultant trajectory as the output of the dashed-boxed system, for which we apply the data-driven predictive control to address the model discrepancy to realize hyper-accurate trajectory tracking. In other words, we *predictively steer the reference*

\* The authors contribute equally to this work. The authors are with the Wisconsin Expeditionary Legged Locomotion (WELL-Lab) at the University of Wisconsin-Madison. Corresponding to xiaobin.xiong@wisc.edu.

The experiment video can be seen here: <https://youtu.be/YM8VbjKEK9A>

trajectory (the input) so that the actual realized one (the output) follows the original target trajectory using a data-driven input-output (DD-IO) model.

We leverage reference-steering via DDPC online to realize precise flying and periodic hopping on the robot PogoX [7], [26] in both simulation and hardware. The original flying and hopping behaviors are realized by model-based controllers, which produce inaccurate tracking due to the model-to-real gap. We show that by utilizing reference-steering, DDPC bridges the gap and significantly improves the tracking accuracy. Moreover, to enable predictive control on hopping with hybrid dynamics, we present an artificial IO data generation process so that a uniform DDPC controller can be used to realize control over hybrid dynamics. The results validate our data-driven reference-steering and indicate a promising system-level data-augmented control design paradigm for complex robotic systems.

## II. RELATED WORK

The literature on data-driven or learning-based methods is very rich. Here, we mainly present the related work on developing data-driven predictive control (DDPC) and its application in robotics. DDPC is a recently popular development on data-driven control in the control community. It utilizes the behavioral system theory [27] to represent dynamical systems via data and then apply the Model Predictive Control (MPC) perspective for control. DeePC is an alternative acronym first used in [19] with its widely accepted formulation. The recent development has been focused on extensions to high-dimensional systems [28], nonlinear dynamical systems [29], and online data updates [30].

DDPC has been applied to robotics systems in the literature with a primary goal of using data to accurately capture unknown robot dynamics. For instance, [31] utilized DDPC to predict accurate quadcopter dynamics. [32] developed a data-driven error model to improve the performance of MPC on manipulators. [33] applied DDPC to better fit a single-rigid-body dynamics model for quadrupedal locomotion. [34] used DDPC to develop a template model for an exoskeleton that can easily learn and adapt to disturbances. In addition to modeling robot dynamics, DDPC can also be used to model disturbance forces and constraints that lack explicit models. [35] has proposed a DDPC-based collaborative control method for modeling constraints of multi-agent robotic systems and the modeled dynamics are incorporated in a distributed trajectory planner.

## III. PRELIMINARIES

We begin by briefly introducing the behavioral system theory and the data-driven predictive control (DDPC) framework [19], [33], [34], [36], [37], which are the main control-theoretic tool utilized in this paper. We also succinctly review our previous model-based control of the robot PogoX [7].

### A. Behavioral System Theory

The fundamental principles of behavioral systems theory [27] offer a mathematically rigorous method to represent an

unknown Linear Time-invariant (LTI) system purely through its measured input-output data. The standard representation of a discrete-time LTI system where the state is denoted as  $x_k \in \mathbb{R}^n$ , the input as  $u_k \in \mathbb{R}^m$  and the output as  $y_k \in \mathbb{R}^p$  for  $k \in \mathbb{Z}_{\geq 0}$  is given by:

$$x_{k+1} = Ax_k + Bu_k, \quad y_k = Cx_k + Du_k, \quad (1)$$

where  $A \in \mathbb{R}^{n \times n}$ ,  $B \in \mathbb{R}^{n \times m}$ ,  $C \in \mathbb{R}^{p \times n}$  and  $D \in \mathbb{R}^{p \times m}$  are the state-space matrices. Given an input trajectory that starts at time  $t_0$  and ends at time  $t_1$ , the sequence  $u_{[t_0, t_1]} := [u_{t_0}^T, u_{t_0+1}^T, \dots, u_{t_1}^T]^T \in \mathbb{R}^{mT}$  has a length  $mT$ . Its Hankel matrix with depth  $L$  can be defined as:

$$\mathcal{H}_L(u_{[t_0, t_1]}) := \begin{bmatrix} u_{t_0} & u_{t_0+1} & \dots & u_{t_1-L+1} \\ u_{t_0+1} & u_{t_0+2} & \dots & u_{t_1-L+2} \\ \vdots & \vdots & \ddots & \vdots \\ u_{t_0+L-1} & u_{t_0+L} & \dots & u_{t_1} \end{bmatrix} \quad (2)$$

*Definition 1:* The input trajectory  $u_{[t_0, t_1]}$  is said to be *persistently exciting* (PE) of order  $L$  if its corresponding Hankel matrix  $\mathcal{H}_L(u_{[t_0, t_1]})$  has full row rank.

*Definition 2:* An input-output pair  $w(t) = (u(t), y(t))$  where  $t \in \mathbb{Z}_{\geq 0}$  and  $w(t) \in \mathbb{R}^{m+p}$  is said to be the *input-output trajectory* of a discrete-time LTI system if for all  $t \in \mathbb{Z}_{\geq 0}$ ,  $\exists x : \mathbb{Z}_{\geq 0} \rightarrow \mathbb{R}^n$  such that 1 holds.

*Theorem 3.1 (Williem's Fundamental Lemma):* For a discrete-time LTI system described in (1), assume  $(A, B)$  is controllable. Let  $(u_{[0, T-1]}, y_{[0, T-1]})$  be a sequence of input-output trajectory with  $T, L \in \mathbb{Z}_{\geq 0}$ . If  $u_{[0, T-1]}$  is persistently exciting of order  $L + n$ , then for a sequence of new input-output trajectory  $(\bar{u}_{[0, L-1]}, \bar{y}_{[0, L-1]})$  there always exists  $g \in \mathbb{R}^{(m+p) \cdot (T-L+1)}$  such that:

$$\begin{bmatrix} \mathcal{H}_L(u_{[0, T-1]}) \\ \mathcal{H}_L(y_{[0, T-1]}) \end{bmatrix} g = \begin{bmatrix} \bar{u}_{[0, L-1]} \\ \bar{y}_{[0, L-1]} \end{bmatrix}. \quad (3)$$

### B. Data-Driven Predictive Control

This section briefly overviews the formulation of data-driven predictive control. As for Williem's Fundamental Lemma, we can assume that  $L = T_{ini} + T_f$ , where  $T_{ini}$  denotes the length of the estimation horizon, and  $T_f$  denotes the prediction horizon. The estimation horizon should be larger than the lag of the system to uniquely determine the initial condition of the unknown LTI system for a given trajectory sequence  $(\bar{u}_{[0, L-1]}, \bar{y}_{[0, L-1]})$ .

*Definition 3:* The *lag*  $\ell$  of a discrete-time LTI system is defined as the smallest integer  $k$  such that the observability matrix  $\mathcal{O}_k$  has rank  $n$ .

*Theorem 3.2:* Let  $T_{ini} \geq \ell$ , and let  $[u_{ini}^T, u^T, y_{ini}^T, y^T]^T$  where  $(u_{ini}, y_{ini}) \in \mathbb{R}^{T_{ini} \cdot (m+p)}$ ,  $(u, y) \in \mathbb{R}^{T_f \cdot (m+p)}$  be a sequence of input-output trajectories. Then there exists a unique state vector  $x_{ini} \in \mathbb{R}^n$  such that:  $y = \mathcal{O}_{T_f} x_{ini} + \mathcal{T}_{T_f} u$ , where  $\mathcal{T}_{T_f}$  is the Toeplitz matrix [19] of the LTI system.

The previous definitions and theorems give rise to the division of Hankel matrices of input-output trajectories into past data for uniquely determining initial conditions and

prediction part in future horizons for control:

$$\mathcal{H}_L(u_{[0,T-1]}) = \begin{bmatrix} U_p \\ U_f \end{bmatrix}, \mathcal{H}_L(y_{[0,T-1]}) = \begin{bmatrix} Y_p \\ Y_f \end{bmatrix}, \quad (4)$$

where  $U_p \in \mathbb{R}^{mT_{\text{ini}} \times (T-L+1)}$ ,  $U_f \in \mathbb{R}^{mT_f \times (T-L+1)}$ ,  $Y_p \in \mathbb{R}^{pT_{\text{ini}} \times (T-L+1)}$  and  $Y_f \in \mathbb{R}^{pT_f \times (T-L+1)}$ , and the subscripts  $p$  and  $f$  denote the data in the past and future, respectively. Then a Data-Driven Predictive Control (DDPC) problem can be formulated as a quadratic program (QP) as in [19]:

$$\begin{aligned} \min_{(g,u,y,\sigma_y)} & \|y - y^{\text{des}}\|_Q^2 + \|u\|_R^2 + \lambda_g \|g\|^2 + \lambda_\sigma \|\sigma_y\|^2, \quad (5) \\ \text{s.t.} & \begin{bmatrix} U_p \\ Y_p \\ U_f \\ Y_f \end{bmatrix} g = \begin{bmatrix} u_{\text{ini}} \\ y_{\text{ini}} - \sigma_y \\ u \\ y \end{bmatrix}, \quad u \in \mathcal{U}, \quad y \in \mathcal{Y}, \end{aligned}$$

where  $Q, R \succ 0$  are weighting matrices for inputs  $u \in \mathbb{R}^{mT_f}$  and outputs  $y \in \mathbb{R}^{pT_f}$ , and  $\lambda_g$  and  $\lambda_\sigma$  act as penalty terms for  $g$  and  $\sigma_y$ . Note that  $\sigma_y$  is commonly introduced to represent measurement noise on the output.  $\mathcal{U}, \mathcal{Y}$  denote the feasible sets of the input and output trajectories.

### C. Flying-Hopping Control of PogoX

Here we briefly review our previous control design [7] for realizing flying-hopping on PogoX. PogoX is a hybrid robot that combines a flying quadrotor with a pogo-stick, which enables terrestrial hopping behaviors. The current version of the robot [26] weighs 3.65kg with a height of 0.53m. It is fully equipped with various sensors and computation power to perform hybrid locomotion in a real-world environment.

Flying Control: Since the core body of the robot is a quadrotor, flying has been directly realized by using its off-the-shelf on-board PX4 flight controller [38], which has cascaded linear control loops to sequentially control angular rate, body orientation, and then linear velocities and linear positions.

Hopping Control: The previous hopping controller is composed of two main components: a vertical energy controller that is responsible for maintaining a certain apex height, and a horizontal velocity controller that aims for velocity tracking. The vertical energy controller is designed as a control Lyapunov Function based Quadratic Program (CLF-QP controller) [7], [39] which stabilizes the vertical mechanical energy to a desired value. The horizontal velocity controller utilizes a step-to-step (S2S) dynamics [5], [7] based stepping control where the nominal hopping gait is optimized from the vertical energy-controlled spring-loaded inverted pendulum (SLIP) model. Readers can refer to [7] for details.

**Remark:** The hopping controller uses the Euler angles and total thrust forces rather than the motor speeds as the control input. This is primarily because the on-board flight controller provides an “off-board” mode in which the users can command the Euler angles and total thrust force, which are then realized by its internal PI/PID controllers, and the users can only adjust the control gains within certain ranges. This layered controller architecture is similar to other robotic systems where the users can command desired torque or joint velocity, which are realized by the proprietary current

and joint controllers from the vendor. In practice, the proprietary controller can have degraded tracking performance if they are not tuned for the target application; in this case, the users cannot modify the proprietary controller part to improve tracking. This also happened to PogoX since the flight controller was designed for quadrotor-flying, thus the proprietary controller oftentimes provides unsatisfying tracking performance. This is one of the motivations of our research here: we aim to steer the reference rather than re-design the existing control architecture.

## IV. REFERENCE-STEERING FOR FLYING

In this section and the next, we describe the reference-steering via DDPC for realizing flying that has continuous dynamics and hopping that has hybrid dynamics. For PogoX which has a “flight controller” of quadrotors, the choice of the tracking trajectories can either be the Cartesian position with yaw angle or the vertical thrust force with orientation, which are in different operation modes (on-board v.s. off-board mode). Switching the modes during locomotion is prohibited due to the hardware. With an aim to use one mode to enable versatile locomotion, we opt to use the “off-board” mode. The reason will be highlighted in the next section.

### A. Flying Dynamics and Controller

Robot Dynamics: The flying dynamics of PogoX is governed by [40]:  $m\ddot{\mathbf{r}} = [0, 0, -mg]^T + \mathbf{R} [0, 0, u_1]^T$ ,  $\mathcal{I}\dot{\omega} = [u_2, u_3, u_4]^T - \omega \times \mathcal{I}\omega$ , where  $u_1$  is the total thrust force,  $u_2, u_3, u_4$  represent the moments generated on the body frame,  $\omega$  represents the angular velocity in body frame, and  $\{\mathbf{r}, \mathbf{R}\}$  represent the position and orientation of the robot in the world frame, respectively. The thrust force and moment generated by the rotor can be modeled as  $\mathcal{F} = k_F \omega_{\text{rotor}}^2$ , where  $k_F$  is constant determined by the propeller, and  $\omega_{\text{rotor}}$  is the rotor velocity. The motor dynamics and its controller are all linear.

Flying Control Design: When designing flying control,  $(\mathbf{r}, \psi)$  is commonly used as the output of trajectory planning and control, where  $\psi$  is the yaw angle of the robot. With an eye to use consistent IO for both flying and hopping, here we choose the Euler angles and total thrust,  $(\psi, \theta, \phi, F_{\text{thrust}})$ , as the output of the “flight controller” in its “off-board” mode. To realize flying in the Cartesian space, we first design a linear PD controller to stabilize the vertical position. The horizontal positions are stabilized by directly calculating the corresponding desired Euler angles. We assume the robot does not turn for simplicity, i.e.  $\psi = 0$ . Since the roll and pitch angles directly relate to the leg angle of the robot, we refer to the model-based controller design as “Height and Leg Angle Controller”, as illustrated in Fig. 2, which realizes both flying and hopping given corresponding desired trajectories.

### B. Reference-Steering via DDPC

We first show that the closed-loop input-output (IO) dynamics of PogoX during flying can be approximated by an LTI system, which rationalizes the application of DDPC

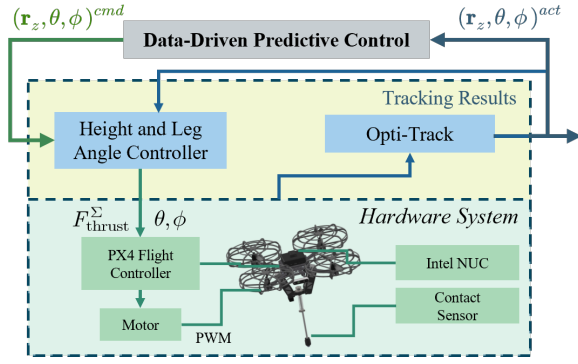


Fig. 2: The control architecture of PogoX.

for reference-steering. We then present a computationally efficient DDPC for realizing online reference-steering.

**Closed-loop IO Dynamics:** During the flying and hopping operation of the robot, it is reasonable to assume: (a) the robot operates with small roll and pitch angles, where  $\cos \phi \approx 1, \sin \phi \approx \phi, \cos \theta \approx 1, \sin \theta \approx \theta$ ; and (b) the cross product term of the angular velocity is relatively small, which yields a linear robot dynamics. The closed-loop IO dynamics of the dashed-box diagram in Fig. 2 is thus simplified to a linear system, as the motor dynamics, and controllers are all linear. Additionally, many unmodeled dynamics and processes on the robot are approximately linear as well, such as the blade flapping, deformation of the limbs, and internal electronic processes. Therefore, it is reasonable to use the IO of the closed-loop system in DDPC for reference-steering.

**Efficient DDPC Formulation:** To apply the DDPC, we first off-line generate sufficient IO data of the closed-loop system to realize flying behaviors to construct the Hankel matrices offline. Then, we use the Hankel matrices to formulate (5) for reference-steering. The offline data may result in large Hankel matrices, which thus prevent real-time computation. Additionally, the sensor noises can lead to infeasibility. To address these problems, we introduce an extension to the PEM-MPC formulation of DDPC [28] to improve the computational efficiency and robustness to sensor noise.

We consider to first solve for the linear predictor  $g$  and the slack variable  $\sigma_y$  of this optimization problem:

$$\min_{g, \sigma_y} \|g\|^2 + \|\sigma_y\|^2, \quad \text{s.t.} \quad \begin{bmatrix} U_p \\ Y_p \\ U_f \end{bmatrix} g = \begin{bmatrix} u_{ini} \\ y_{ini} + \sigma_y \\ u \end{bmatrix}, \quad (6)$$

the solution of which is ( $^\dagger$  represents the pseudo-inverse):

$$\begin{bmatrix} g \\ \sigma_y \end{bmatrix} = \begin{bmatrix} U_p & 0 \\ Y_p & I \\ U_f & 0 \end{bmatrix}^\dagger \begin{bmatrix} u_{ini} \\ y_{ini} \\ u \end{bmatrix}. \quad \text{Let } \mathcal{G} = \begin{bmatrix} Y_f & 0 \\ 0 & I \end{bmatrix} \begin{bmatrix} U_p & 0 \\ Y_p & I \\ U_f & 0 \end{bmatrix}^\dagger.$$

The reference-steering problem can thus be formulated as:

$$\begin{aligned} \min_{u, y, \sigma_y} & \|y - y^{\text{des}}\|_Q^2 + \|u\|_R^2 + \lambda_\sigma \|\sigma_y\|^2, \quad (\text{DDPC-Flying}) \\ \text{s.t.} & \begin{bmatrix} y \\ \sigma_y \end{bmatrix} = \mathcal{G} \begin{bmatrix} u_{ini} \\ y_{ini} \\ u \end{bmatrix}, \quad u \in \mathcal{U}, y \in \mathcal{Y}, \end{aligned}$$

where  $y^{\text{des}}$  is the desired trajectory,  $u$  and  $y$  are commanded trajectory and realized trajectory.  $\mathcal{G}$  is computed offline, and solving this QP online yields the control input  $u$  to steer the realized output  $y$  towards the desired one  $y^{\text{des}}$ .

## V. REFERENCE-STEERING FOR PERIODIC HOPPING

We now present our approach of using reference-steering via DDPC to enable periodic hopping locomotion on PogoX. Its hybrid dynamics presents a severe challenge from both a theoretical and a practical perspective on using the continuous predictive control approach for trajectory stabilization via either MPC or DDPC. We thus innovate an "artificial IO" trajectory generation process on the ground phase to enable continuous predictive control via DDPC in the aerial phase.

### A. Hybrid Dynamics and Hopping Control

**Hybrid Dynamics of Hopping:** The dynamics of hopping is hybrid, as it has two different phases: an aerial phase and a ground phase. The dynamics in the aerial phase are identical to the quadrotor dynamics. On the ground, the robot is subjected to ground reaction force due to contact, and we assume the foot does not slip during contact, which yields a holonomic constraint. The dynamics in the ground phase can thus be described by a differential-algebraic equation. The transition from the aerial phase to the ground phase is governed by a discrete impact map, while the transition from the ground phase to the aerial phase occurs smoothly. The complete hybrid dynamics description can be seen in [7].

**Hopping Control Design:** To enable dynamic hopping behaviors with consistent IO of flying, we modify our previous hopping control in [7]. Instead of stabilizing the energy to the desired one in the aerial phase, we directly track the vertical height of the robot. The leg angles, equivalent to the roll and pitch angles of the robot body, are used for control which is the same as the previous one in [7]. The desired vertical trajectory is optimized using the robot dynamics for vertical hopping. The leg angles can perturb and stabilize the vertical hopping into hopping with different horizontal velocity via step-to-step (S2S) dynamics based control [5]. Since the ground phase is very short and abrupt, we simply let the desired output values be the current measured ones, which can preserve the natural dynamics of the spring leg.

**Choice of IO:** The alternative choice of IO can be the vertical height plus the horizontal position of the robot. It may be able to produce dynamic hopping if the horizontal desired trajectory is well-designed or optimized, which thus requires additional trajectory optimization to realize hopping with different horizontal velocities. While for hopping with leg angle as outputs, variable horizontal velocities can be produced directly [4], [5] without additional trajectory optimization. Therefore, in our application, we select the vertical height and body orientation as outputs. This is the reason we use these outputs for flying rather than the Cartesian positions.

### B. Reference-Steering via DDPC

**Artificial IO Generation:** This model-based hopping control design can produce dynamic hopping. Yet, due to the model

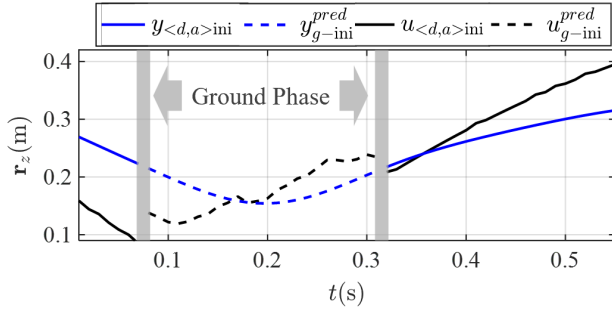


Fig. 3: Illustration of artificial ground phase trajectory generation, where  $u_{<d,a>ini}$  and  $y_{<d,a>ini}$  represent the collected input and output data, and  $u_{g-ini}^{pred}$ ,  $y_{g-ini}^{pred}$  represent the generated IO data of the ground phase.

discrepancy and the tracking errors on the desired outputs, the realized hopping can have errors in its hopping height and velocity. We then apply the reference-steering with DDPC to improve the trajectory tracking results. Since the hopping dynamics is hybrid, the Hankel matrix built for flying tasks cannot be directly applied to hopping, and the IO data of hopping cannot be used for hopping since they are from two different dynamics. Fortunately, the IO data of the aerial phase still represents the dynamics of the aerial phase, which is the phase we apply control. Therefore, we focus on generating artificial IO data in the ground phase that represents the aerial dynamics. With artificial IO data in the ground phase and the real IO data in the aerial phase, we have continuous IO trajectories of the hopping behavior, where the IO data only represents the aerial dynamics. We can then apply DDPC in the aerial phase that has the approximately linear IO dynamics shown in the previous section.

Then our goal is to design  $u_{g-ini}$ ,  $y_{g-ini}$  that artificially represent the ground trajectories using aerial dynamics. For each hopping, we formulate this QP problem:

$$\begin{aligned} \min_{u_{g-ini}, y_{g-ini}, g} & \|g\|^2, \\ \text{s.t. } \begin{bmatrix} U_p \\ Y_p \end{bmatrix} g &= \begin{bmatrix} u_{<d,g,a>ini} \\ y_{<d,g,a>ini} \end{bmatrix}, u \in \mathcal{U}, y \in \mathcal{Y}, \end{aligned} \quad (7)$$

where  $U_p, Y_p$  are the Hankel matrices representing the aerial dynamics, and the subscripts  $d, g, a$  denote the descending, ground, and ascending phases, respectively. Fig. 3 illustrates that by selecting an appropriate length for the flying initial IO trajectories  $T_{d-ini}$  and  $T_{a-ini}$ , the artificial ground IO trajectories  $u_{g-ini}$ ,  $y_{g-ini}$  can be uniquely determined. If we collect  $n$  trajectories from  $n$  hopping cycles, a new Hankel matrix, which incorporates artificial ground phase initial trajectories, can be expressed as  $H_G = [H_{G1} | H_{G2} | \dots | H_{Gn}]$  where  $H_{Gk}$  is the Hankel matrix of  $k$ -th trajectory  $[u_{d-ini}^k, u_{g-ini}^k, u_{a-ini}^k, y_{d-ini}^k, y_{g-ini}^k, y_{a-ini}^k]$  with  $k \in \mathbb{Z}_{[1,n]}$ .

**DDPC for Periodic Hopping:** With a relatively short and fixed ground phase, we can infer that the length of the initial trajectories exceeds that of the ground phase. We thus implicitly assume the length of the initial trajectories before and after the ground phase is longer than the system lag

III-B such that system states at lift-off and touch-down are uniquely determined. The DDPC problem, which includes the ground phase, can then be described as follows:

$$\begin{aligned} \min_{u, y, \sigma_y} & \|y - y^{des}\|_Q^2 + \|u\|_R^2 + \lambda_\sigma \|\sigma_y\|^2, \quad (\text{DDPC-Hopping}) \\ \text{s.t. } \begin{bmatrix} y \\ \sigma_y \end{bmatrix} &= \mathcal{G} \begin{bmatrix} u_{<d,g,a>ini} \\ y_{<d,g,a>ini} \\ u \end{bmatrix}, u \in \mathcal{U}, y \in \mathcal{Y}. \end{aligned}$$

This formulation is then uniform with that in (DDPC-Flying). Similarly,  $\mathcal{G}$  is computed offline, and then this QP problem can be computed online in real time.

## VI. EVALUATION

In this section, we will present the results of our approach for realizing hyper-accurate trajectory tracking on the robot PogoX in both simulation and hardware. The hardware control is realized in ROS environment. For indoor testing, the actual trajectories are measured by an Opti-track system that is composed of 12 Prime-13 cameras. The proposed DDPC problems are implemented in C++, and the corresponding QPs are solved using OSQP [41] with warm-starting at 200 Hz, which matches the computation frequency of the reference trajectory planners and the low-level controllers. The simulation is implemented using MATLAB ODE function with event-triggering, where the control part is set up similarly to match that of the hardware.

To mitigate the potential of non-persistent excitation in the input signal, which is required by 3.1, we adopt the approach in [31] by applying a small scaled pseudo-random binary sequence (PRBS) noise to the input signals while minimizing its influence on the control quality.

The previous version of PogoX [26] has a thrust-to-weight ratio (TWR) estimated at  $\sim 0.82$ . To enable flying in the work, we take the batteries off the robot to have a TWR slightly over 1. All the flying and hopping data for the Hankel matrix construction were collected under this condition.

### A. Reference-steering for Flying

We first present the results of applying reference-steering to the flying of PogoX. The input data used to construct the Hankel matrices is collected on the desired  $z$  and  $\theta$  trajectories that are mapped from RC control inputs. The output data comes from the direct measurement of the robot states. This data collection intends to capture the general IO dynamics in the air rather than those of specific tasks for following certain trajectories. The lower-level controller and its feedback gains during data collection and experiment remain the same. The robot is then commanded to follow an ellipsoid trajectory in its sagittal plane where the desired  $x$ -direction position is transformed to the desired leg angle via a PD controller. Since the reference trajectories are designed to be relatively short, the initial trajectory length is set to 20 to sufficiently account for system lag, while the prediction horizon is kept relatively small to 15 to enhance computational efficiency.

**Simulation:** The height and leg angle reference tracking performance can be seen in Fig. 4. To challenge the controller,

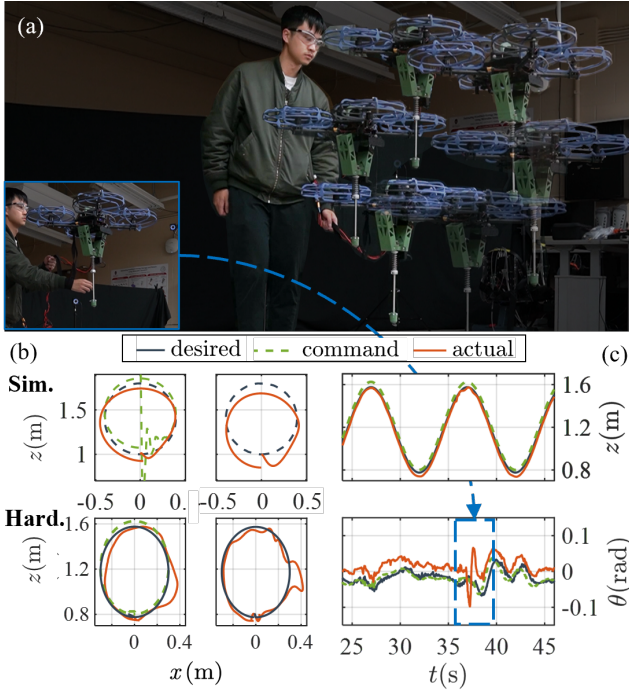


Fig. 4: Results on flying: (a) depicts PogoX in flying and disturbance injection; (b) compares the quality of trajectory tracking in  $x-z$  plane; and (c) demonstrates how DDPC responds to disturbances in both height and leg angle tracking.

the robot mass in the controller is set to be 200g less than its real value. As shown in the  $x-z$  plot, the quadrotor failed to follow the correct trajectory due to the modeling error, and reference-steering via DDPC steers it back to the desired height. Note that to enhance tracking in the  $x$  direction, one can always add  $x$  position in the output to formulate the dynamics with input as reference  $\theta$  and  $x$ .

**Hardware:** The same controller and desired trajectory are used on the hardware with additional disturbance forces applied to the robot leg to disrupt both height and leg angle tracking. The comparison of the robustness of the controller, with and without DDPC, is shown in Fig. 4. Despite we cannot quantitatively measure the applied disturbance for each experiment, DDPC is showing better performance for both regular tracking and disturbance rejection.

### B. Reference-steering for Periodic Hopping

We now present the results of the periodic hopping behaviors. During both the data collection and experimental evaluation, the robot is commanded to perform periodic hopping, aiming to realize a desired apex height and a target horizontal velocity. To ensure adequate coverage of both the descent and ascent phases during the aerial stage in predictive control, the prediction horizon is set to 25, and the initial trajectory length is maintained at 20. This allows the horizon to encompass a complete hopping cycle.

**Simulation:** Fig. 5 illustrates the hopping behavior performance in simulation, both with and without DDPC. It is evident that with DDPC, height tracking is significantly

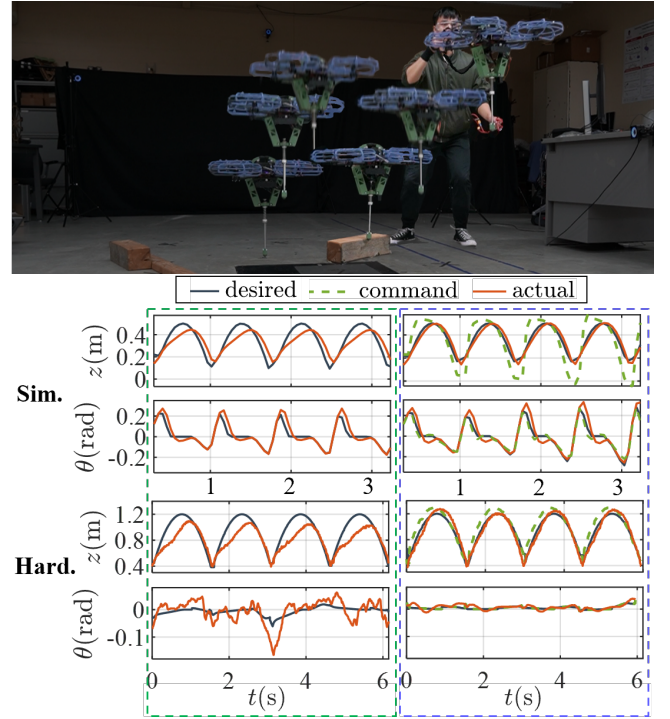


Fig. 5: Results on periodic hopping: trajectories without reference-steering (green) and with reference-steering (blue).

improved, while the leg angles maintain sufficiently accurate tracking performance. With DDPC, the reference trajectories are steered to higher targets to account for the tracking inaccuracies during lift-off.

**Hardware:** The height and leg angle reference tracking performance can be seen in Fig. 5. Similar to the simulation results, the height and leg angle tracking have been significantly improved, leading to better overall stabilization of step-to-step (S2S) dynamics and hopping behaviors. The steered output demonstrates that DDPC can predict the mismatch between the reference and actual trajectories, adjusting the reference trajectory to better track the desired trajectories.

## VII. CONCLUSION AND FUTURE WORK

In conclusion, we propose a reference-steering approach via Data-Driven Predictive Control to improve trajectory tracking on robotic locomotion. We evaluated this approach both in simulation and on the hardware of the robot PogoX to realize both flying and periodic hopping locomotion. The results show significant tracking improvement for both locomotion behaviors, indicating a promising data-driven control approach that is augmented to the original model-based control design for general robotic systems.

Our future work will be on extensions to nonlinear closed-loop dynamics and online updating of the Hankel matrices to build data-driven models that can adapt to system dynamics going through changes, e.g. locomotion over unknown and varying terrains. Additionally, we are interested in designing data-driven modeling and predictive control with augmentation to the model-based control and state estimation for general robotics tasks.

## REFERENCES

- [1] R. Mahony, V. Kumar, and P. Corke, "Multirotor aerial vehicles: Modeling, estimation, and control of quadrotor," *IEEE robotics & automation magazine*, vol. 19, no. 3, pp. 20–32, 2012.
- [2] M. Maaruf, M. S. Mahmoud, and A. Ma'arif, "A survey of control methods for quadrotor uav," *International Journal of Robotics and Control Systems*, vol. 2, no. 4, pp. 652–665, 2022.
- [3] P. M. Wensing, M. Posa, Y. Hu, A. Escande, N. Mansard, and A. Del Prete, "Optimization-based control for dynamic legged robots," *IEEE Transactions on Robotics*, 2023.
- [4] M. H. Raibert, *Legged robots that balance*. MIT press, 1986.
- [5] X. Xiong and A. Ames, "3-d underactuated bipedal walking via h-hip based gait synthesis and stepping stabilization," *IEEE Transactions on Robotics*, vol. 38, no. 4, pp. 2405–2425, 2022.
- [6] B. Landry, R. Deits, P. R. Florence, and R. Tedrake, "Aggressive quadrotor flight through cluttered environments using mixed integer programming," in *2016 IEEE international conference on robotics and automation (ICRA)*. IEEE, 2016, pp. 1469–1475.
- [7] Y. Wang, J. Kang, Z. Chen, and X. Xiong, "Terrestrial locomotion of pogo: From hardware design to energy shaping and step-to-step dynamics based control," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*, 2024, pp. 3419–3425.
- [8] G. Bledt, M. J. Powell, B. Katz, J. Di Carlo, P. M. Wensing, and S. Kim, "MIT cheetah 3: Design and control of a robust, dynamic quadruped robot," in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 2245–2252, ISSN: 2153-0866.
- [9] Y. Ding, C. Khazoom, M. Chignoli, and S. Kim, "Orientation-aware model predictive control with footstep adaptation for dynamic humanoid walking," in *2022 IEEE-RAS 21st International Conference on Humanoid Robots (Humanoids)*. IEEE, 2022, pp. 299–305.
- [10] H. Li, T. Zhang, W. Yu, and P. M. Wensing, "Versatile real-time motion synthesis via kino-dynamic MPC with hybrid-systems DDP." [Online]. Available: <http://arxiv.org/abs/2209.14138>
- [11] B. Siciliano, "Springer handbook of robotics," *Springer-Verlag google schola*, vol. 2, pp. 15–35, 2008.
- [12] K. Lynch, *Modern Robotics*. Cambridge University Press, 2017.
- [13] G. Shi, X. Shi, M. O'Connell, R. Yu, K. Azzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural lander: Stable drone landing control using learned dynamics," in *2019 international conference on robotics and automation (icra)*. IEEE, 2019, pp. 9784–9790.
- [14] I. Markovsky and P. Rapisarda, "Data-driven simulation and control," *International Journal of Control*, vol. 81, no. 12, pp. 1946–1959, 2008.
- [15] M. Dai, X. Xiong, J. Lee, and A. D. Ames, "Data-driven adaptation for robust bipedal locomotion with step-to-step dynamics," in *2023 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*. IEEE, 2023, pp. 8574–8581.
- [16] Z. Li, X. B. Peng, P. Abbeel, S. Levine, G. Berseth, and K. Sreenath, "Reinforcement learning for versatile, dynamic, and robust bipedal locomotion control," *arXiv preprint arXiv:2401.16889*, 2024.
- [17] B. Bianchini, M. Halm, and M. Posa, "Simultaneous learning of contact and continuous dynamics," in *Conference on Robot Learning*. PMLR, 2023, pp. 3966–3978.
- [18] R. Ferede, C. De Wagter, D. Izzo, and G. C. De Croon, "End-to-end reinforcement learning for time-optimal quadcopter flight," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 6172–6177.
- [19] J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control: In the shallows of the deepc," in *2019 18th European Control Conference (ECC)*, 2019, pp. 307–312.
- [20] Z. Su, X. Huang, D. Ordoñez-Apaez, Y. Li, Z. Li, Q. Liao, G. Turrissi, M. Pontil, C. Semini, Y. Wu *et al.*, "Leveraging symmetry in rl-based legged locomotion control," *arXiv preprint arXiv:2403.17320*, 2024.
- [21] G. Bellegarda, C. Nguyen, and Q. Nguyen, "Robust quadruped jumping via deep reinforcement learning," *Robotics and Autonomous Systems*, p. 104799, 2024.
- [22] Y.-M. Chen, H. Bui, and M. Posa, "Reinforcement learning for reduced-order models of legged robots," in *2024 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 2024, pp. 5801–5807.
- [23] A. H. Chang, C. M. Hubicki, J. J. Aguilar, D. I. Goldman, A. D. Ames, and P. A. Vela, "Learning terrain dynamics: A gaussian process modeling and optimal control adaptation framework applied to robotic jumping," *IEEE Transactions on Control Systems Technology*, vol. 29, no. 4, pp. 1581–1596, 2020.
- [24] T. Salzmann, E. Kaufmann, J. Arrizabalaga, M. Pavone, D. Scaramuzza, and M. Ryll, "Real-time neural mpc: Deep learning model predictive control for quadrotors and agile robotic platforms," *IEEE Robotics and Automation Letters*, vol. 8, no. 4, pp. 2397–2404, 2023.
- [25] Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, and V. Sindhwani, "Data efficient reinforcement learning for legged robots," in *Conference on Robot Learning*. PMLR, 2020, pp. 1–10.
- [26] J. Kang, Y. Wang, and X. Xiong, "Fast decentralized state estimation for legged robot locomotion via ekf and mhe," *IEEE Robotics and Automation Letters*, *Under Review*, 2024.
- [27] J. C. Willems, P. Rapisarda, I. Markovsky, and B. L. De Moor, "A note on persistency of excitation," *Systems and Control Letters*, vol. 54, no. 4, pp. 325–329, 2005. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167691104001434>
- [28] L. Huang, J. Coulson, J. Lygeros, and F. Dörfler, "Data-enabled predictive control for grid-connected power converters," in *2019 IEEE 58th Conference on Decision and Control (CDC)*, 2019, pp. 8130–8135.
- [29] Y. Lian and C. N. Jones, "Nonlinear data-enabled prediction and control," in *Proceedings of the 3rd Conference on Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, A. Jadbabaie, J. Lygeros, G. J. Pappas, P. A. Parrilo, B. Recht, C. J. Tomlin, and M. N. Zeilinger, Eds., vol. 144. PMLR, 07 – 08 June 2021, pp. 523–534. [Online]. Available: <https://proceedings.mlr.press/v144/lian21a.html>
- [30] S. Baros, C.-Y. Chang, G. E. Colón-Reyes, and A. Bernstein, "Online data-enabled predictive control," *Automatica*, vol. 138, p. 109926, 2022. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0005109821004507>
- [31] E. Elokda, J. Coulson, P. Beuchat, J. Lygeros, and F. Dörfler, "Data-enabled predictive control for quadcopters," *International Journal of Robust and Nonlinear Control*, vol. 31, 07 2021.
- [32] A. Carron, E. Arcari, M. Wermelinger, L. Hewing, M. Hutter, and M. N. Zeilinger, "Data-driven model predictive control for trajectory tracking with a robotic arm," *IEEE Robotics and Automation Letters*, vol. 4, no. 4, pp. 3758–3765, 2019.
- [33] R. T. Fawcett, K. Afsari, A. D. Ames, and K. A. Hamed, "Toward a data-driven template model for quadrupedal locomotion," *IEEE Robotics and Automation Letters*, vol. 7, no. 3, pp. 7636–7643, 2022.
- [34] K. Li, J. Kim, X. Xiong, K. A. Hamed, Y. Yue, and A. D. Ames, "Data-driven predictive control for robust exoskeleton locomotion," in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, *To appear*, 2024.
- [35] R. T. Fawcett, L. Amanzadeh, J. Kim, A. D. Ames, and K. A. Hamed, "Distributed data-driven predictive control for multi-agent collaborative legged locomotion," in *2023 IEEE International Conference on Robotics and Automation (ICRA)*, 2023, pp. 9924–9930.
- [36] S. Baros, C.-Y. Chang, G. E. Colon-Reyes, and A. Bernstein, "Online data-enabled predictive control," *Automatica*, vol. 138, p. 109926, 2022.
- [37] J. Berberich, J. Köhler, M. A. Müller, and F. Allgöwer, "Data-driven model predictive control with stability and robustness guarantees," *IEEE Transactions on Automatic Control*, vol. 66, no. 4, pp. 1702–1717, 2021.
- [38] L. Meier, D. Honegger, and M. Pollefeys, "Px4: A node-based multithreaded open source robotics framework for deeply embedded platforms," in *2015 IEEE International Conference on Robotics and Automation (ICRA)*, 2015, pp. 6235–6240.
- [39] A. D. Ames, K. Galloway, K. Sreenath, and J. W. Grizzle, "Rapidly exponentially stabilizing control lyapunov functions and hybrid zero dynamics," *IEEE Transactions on Automatic Control*, vol. 59, no. 4, pp. 876–891, 2014.
- [40] F. Sabatino, "Quadrotor control: modeling, nonlinear control design, and simulation," 2015. [Online]. Available: <https://api.semanticscholar.org/CorpusID:61413561>
- [41] B. Stellato, G. Banjac, P. Goulart, A. Bemporad, and S. Boyd, "Osqp: An operator splitting solver for quadratic programs," *Mathematical Programming Computation*, vol. 12, no. 4, pp. 637–672, 2020.