Continual Learning and Lifting of Koopman Dynamics for Linear Control of Legged Robots

Feihan Li FEIHANL@ANDREW.CMU.EDU
Abulikemu Abuduweili ABULIKEA@ANDREW.CMU.EDU
Yifan Sun YIFANSU2@ANDREW.CMU.EDU
Rui Chen RUIC3@ANDREW.CMU.EDU
Weiye Zhao WEIYEZHA@ANDREW.CMU.EDU
Changliu Liu CLIU6@ANDREW.CMU.EDU
Robotics Institute, Carnegie Mellon University, Pittsburgh, PA 15213, USA

Editors: N. Ozay, L. Balzano, D. Panagou, A. Abate

Abstract

The control of legged robots, particularly humanoid and quadruped robots, presents significant challenges due to their high-dimensional and nonlinear dynamics. While linear systems can be effectively controlled using methods like Model Predictive Control (MPC), the control of nonlinear systems remains complex. One promising solution is the Koopman Operator, which approximates nonlinear dynamics with a linear model, enabling the use of proven linear control techniques. However, achieving accurate linearization through data-driven methods is difficult due to approximation error and domain shifts. These challenges restrict the scalability of Koopman-based approaches. This paper addresses these challenges by proposing Incremental Koopman algorithm designed to iteratively refine Koopman dynamics for high-dimensional legged robots. The key idea is to progressively expand the dataset and latent space dimension, enabling the learned Koopman dynamics to converge towards true system dynamics. Theoretical analysis shows that the linear approximation error of our method converges monotonically. Experimental results demonstrate that our supriority on robots like Unitree G1, H1, A1, Go2, and ANYmal D, across various terrains. This is the first work to apply linearized whole body dynamics with Koopman Operator for locomotion control of high-dimensional legged robots, providing a scalable model-based control solution. The code can be found at: https://github.com/intelligent-control-lab/Incremental-Koopman.

Keywords: Koopman Operator, Model Predictive Control, Legged Robots, Continual Learning

1. Introduction

Control of legged robots has attracted growing interest, driven by the potential of humanoid and quadruped robots in real-world applications. However, the task remains challenging due to the high-dimensional, nonlinear dynamics of legged locomotion. Control strategies for nonlinear systems face a trade-off between controller complexity and modeling effort, broadly categorized as model-free or model-based. Model-free methods directly derive the control law without explicitly modeling the system dynamics. Reinforcement learning (RL) is the representative example, demonstrating notable success in controlling legged robots in real-world applications Song et al. (2021); Haarnoja et al. (2018); Zhao et al.; Lee et al. (2020); He et al. (2024); Fu et al. (2024); Luo et al. (2023). Despite their strong performance, model-free control laws are often highly task-specific and require retraining or fine-tuning when adapting to new tasks.

Model-based approaches decouple the modeling of system dynamics from the realization of task objectives, enabling more efficient adaptation to new tasks. The models used in these approaches

can be catagorized as: original nonlinear models, locally linearized models, and globally linearized models, with increasing modeling complexity and decreasing control complexity. For direct control of nonlinear models, Yi et al. (2024) introduced sampling-based MPC for legged robots, while Kazemi et al. (2013) applied robust backstepping control to quadruped robots. Local linearization methods, iLQR and NMPC often combine analytical Sotaro Katayama and Tazaki (2023); Zhu et al. (2024); Zhang et al. (2024) or learned models Nagabandi et al. (2017); Liu et al. (2023) with complex controllers, requiring extensive domain expertise. A promising alternative to these approaches is employing Koopman Operator Theory to construct globally linearized models of legged systems and enabling the use of a simpler controller, e.g., linear MPC. Although Koopman-based methods have proven effective in motion planning Kim et al. (2024) and gait prediction Krolicki et al. (2022), no prior work has directly modeled the whole body dynamics of legged robots with Koopman Operator.

A Koopman Operator can be derived analytically or approximated through data-driven approaches. While analytical approaches are extremely challenging for high-dimensional nonlinear systems like legged robots, our method adopts a data-driven approach that reduces reliance on domain expertise and enhances generalizability. Recent works have applied data-driven approaches to obtain linear models for high-level motion planning Kim et al. (2024); Lyu et al. (2023). However, these methods typically depend on end-to-end optimization aimed at specific task objectives, which can limit their broader applicability. In contrast, Shi and Meng (2022); Korda and Mezić (2018b); Mamakoukas et al. (2021) have focused on obtaining accurate Koopman approximations for low-level control, but their applicability is limited to low-dimensional systems Shi et al. (2024). It remains challenging to apply Koopman Operator to linearize high-dimensional nonlinear systems due to (a) imperfect approximations and (b) domain shifts arising from limited state transition data that only covers a subspace of the entire state space. Such approximation errors make the system sensitive to inaccurately modeled dynamics, while domain shifts cause failures under perturbations.

To address these challenges, we propose an incremental Koopman algorithm for high-dimensional legged robots that continually learn and lift Koopman dynamics. Our approach gradually expands both the training dataset and the latent space dimension. The core idea is that this expansion progressively spans a state space where refined Koopman dynamics reduce linear approximation errors, ultimately converging to the true Koopman operator, as supported by our theoretical analysis. Experimental results demonstrate that our approach reduces linearization error within a few iterations and enables effective locomotion control for legged robots using Model Predictive Control (MPC). Our key contributions are summarized as follows:

1. We propose an iterative continual learning algorithm to mitigate linear approximation errors for Koopman dynamics, supported by a theoretical guarantee of convergence.
2. Our approach is the first work to implement locomotion control for high-dimensional legged robots using linearized whole body dynamics with Koopman Operator.
3. Compared to traditional NMPC or RL mehods, our approach offers a scalable way to accurately model the dynamics of high-dimensional systems without requiring extensive expert knowledge, while maintaining generalizability across tasks and systems.

2. Problem Formulation

2.1. Preliminary: Koopman Operator Theory

Koopman Operator Theory offers a powerful framework for analyzing nonlinear dynamical systems by mapping them into equivalent linear systems in an infinite-dimensional latent space. Consider

the discrete-time nonlinear autonomous system $s_{t+1} = f(s_t)$, where $s_t \in \mathcal{S}$ represents the state. Koopman operator theory introduces the embedding function $\phi : \mathcal{S} \to \mathcal{O}$ to map the original state space $\mathcal{S}$ to an infinite dimensional latent space $\mathcal{O}$. In this latent space, the dynamics become linear through the Koopman Operator $\mathcal{K}$:

$$\mathcal{K}\phi(s) = \phi(f(s)) \tag{1}$$

Consider a non-autonomous dynamical system with control input $u_t \in \mathcal{U} \subset \mathbb{R}^{m'}$ and system state $x_t \in \mathcal{X} \subset \mathbb{R}^{n'}$, given by $x_{t+1} = f(x_t, u_t)$. The Koopman Operator transforms the system dynamics into a linear form.

$$\mathcal{K}\phi(x_t, u_t) = \phi(f(x_t, u_t)) = \phi(x_{t+1}) \tag{2}$$

In practice, we approximate the Koopman operator by constraining the latent space to a finite-dimensional vector space. For a non-autonomous system, a common strategy is to define the embedding function as $\phi(x_t, u_t) = [g(x_t); u_t]$, where $g : \mathbb{R}^{n'} \to \mathbb{R}^n$ is a state embedding function. The Koopman operator $\mathcal{K}$ can be approximated by a matrix representation $K = \begin{bmatrix} A \in \mathbb{R}^{n\times n} & B \in \mathbb{R}^{n\times m'} \\ C \in \mathbb{R}^{m'\times n} & D \in \mathbb{R}^{m'\times m'} \end{bmatrix}$. In conjunction with (2), we obtain a linear model in the lifted space with the state evolution:

$$g(x_{t+1}) = Ag(x_t) + Bu_t \tag{3}$$

To preserve state information with clear semantics and avoid the case of degeneration Shi and Meng (2022), we concatenate the original state with the neural network embedding to define the latent state z_t as $z_t = g(x_t) = \left[x_t, g'(x_t)\right]^\top$, where $g' : \mathbb{R}^{n'} \to \mathbb{R}^{n-n'}$ is a parameterized neural network. This approach allows us to retrieve the original state using a linear transformation.

$$x_{t+1} = Pz_{t+1} = Pg(x_{t+1}),\ P = \left[I_{n'\times n'}, \mathbf{0}_{n'\times(n-n')}\right] \in \mathbb{R}^{n'\times n} \tag{4}$$

This formulation supports solving state-constrained control problems (e.g., collision avoidance) by directly transforming the constraints on x to constraints on z without compromising the lifted system's linear properties. Additionally, we use an end-to-end training approach for both the embedding function and the Koopman operator, parameterized as $\mathcal{T} \doteq (g, A, B)$.

2.2. Linear Model Predictive Control

With the linear model from the Koopman operator, in our high-dimensional legged robot control tasks, we employ a linear MPC algorithm as the controller π_{mpc}, thus the problem can be easily solved by Quadratic Programming (QP). The optimization formulation at time step t is as follows:

$$\begin{aligned} \min_{u_{t:t+H-1}} \quad & \|Pz_{t:t+H-1} - x^*_{t:t+H-1}\|^2_Q + \|u_{t:t+H-1}\|^2_R + \|Pz_{t+H} - x^*_{t+H}\|^2_F \\ s.t. \quad & z_{t+k+1} = Az_{t+k} + Bu_{t+k},\ \ u_{t+k} \in [u_{min}, u_{max}],\ \ \forall k = 0, \cdots, H-1 \end{aligned} \tag{5}$$

where H denotes the horizon length, $x^*_{t:t+H}$ represents the reference trajectory, Q, R, F are cost matrices and $\|x\|^2_Q$ is defined as x^TQx. The range $[u_{min}, u_{max}]$ specifies the valid interval for control inputs. The first constraint ensures adherence to the learned dynamics in the latent space, while the second constraint bounds control signals within the valid region. It should be noted that, unlike model-free methods, the decomposition of dynamics and controller makes full-body reference tracking essential for maintaining stable and reasonable movement patterns, as the MPC controller itself does not imply gait information.

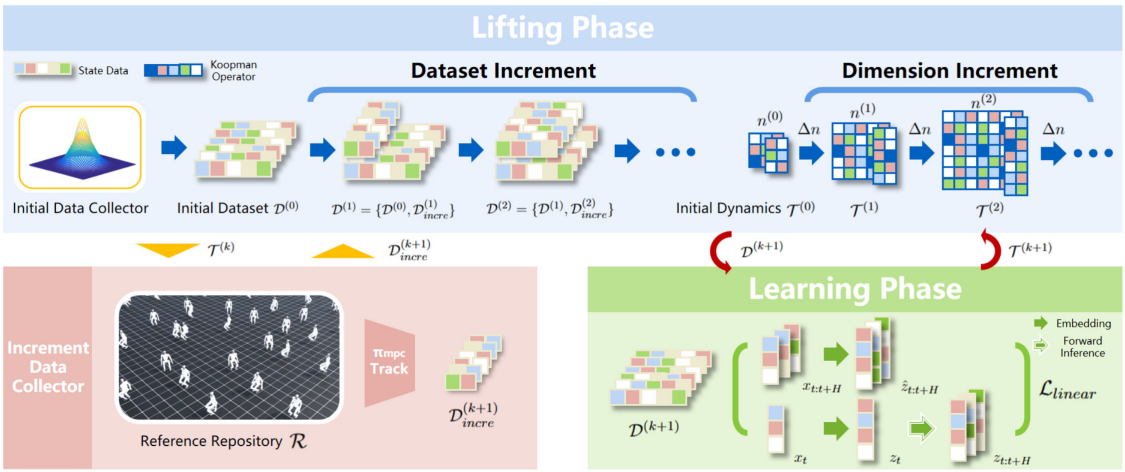


Figure 1: Overview of Incremental Koopman Algorithm

2.3. Legged Robot System

As demonstrated in Wieber et al. (2016), legged robot systems inherently possess the following characteristics: (i) discontinuities resulting from mode transitions, (ii) sensitivity to noisy control inputs while maintaining balance, and (iii) failure situations that are often irrecoverable. Existing Koopman-based methods focus mainly on low-dimensional continuous systems with fixed contact modes Shi and Meng (2022), where the dynamics are continuous and easier to fit. However, when applied to high-dimensional hybrid legged systems, the complex and diverse contact modes introduce significant discontinuities, greatly complicating the modeling process. Moreover, these existing Koopman methods, which exhibit considerable approximation errors and limited latent subspace Han et al. (2023), render hybrid legged systems prone to irrecoverable failures caused by unmodeled dynamics and external perturbations. To overcome these challenges, we will introduce an algorithm that accurately and robustly models dynamics through continual learning and lifting, effectively expanding a resilient latent subspace.

3. Incremental Koopman Algorithm

As shown in Figure 1, the pipeline consists of two data collectors and two phases. After the initial data collection and dynamics learning, we iteratively alternate between the lifting phase, where the dataset is expanded by the increment data collector and the latent space dimension is increased, and the learning phase, during which dynamics are retrained based on the updated dataset and dimension.

3.1. Initialization and Increment of Dataset

Initial Data Collector. Unlike low-dimensional continuous systems explored in Shi and Meng (2022), the high-dimensional hybrid dynamics of legged robots need training data with reasonable gait and consistent contact modes to form a meaningful latent subspace. Thus, we use an initial data collector (e.g., an RL policy or tele-operation) to generate dynamically feasible trajectories with reasonable movement patterns for the construction of initial dataset $\mathcal{D}^{(0)}$.

Increment Data Collector. Our method enables robust tracking within the reference repository $\mathcal{R}$, where references may be dynamically feasible or infeasible, with gaits and contact modes

consistent with $\mathcal{D}^{(0)}$. While $\mathcal{D}^{(0)}$ lays the foundation for the latent subspace, learning from limited samples—especially from a "good behavioral collector"—is insufficient. Models may still exhibit significant approximation errors in near-failure or failure scenarios, leading to ineffective tracking. Thus we use the MPC controller in Section 2.2 to track references in $\mathcal{R}$, collecting failed tracking data $\mathcal{D}_{incre}$ to naturally expand the dataset, thereby enhancing the robustness of the latent subspace.

Notably, our algorithm can be applied to any eligible initial data collector and reference repository $\mathcal{R}$. In this work, within a deterministic environment, we use a Proximal Policy Optimization (PPO Schulman et al. (2017)) policy to build $\mathcal{D}^{(0)}$ and the initial reference repository $\mathcal{R}_{init}$, then add uniform noise in the range [-0.05, 0.05] to all references, generating the dynamically infeasible $\mathcal{R}$.

3.2. Training with Given Data

Based on the base Koopman framework introduced in Section 2.1, our primary goal is to learn the embedding function g and the Koopman operators A, B end-to-end. To achieve this, we employ a discounted k-step prediction loss to enhance long-horizon prediction capabilities which is crucial for MPC control. Given a set of trajectory data $\{x_{t:t+H}; u_{t:t+H-1}\}$, real latent trajectories can be computed as $\{z_{t:t+H} = g(x_{t:t+H})\}$. At the same time, predicted latent trajectories can be obtained through inference: $\{\hat{z}_{t:t+H} | \hat{z}_t = z_t, \hat{z}_{t+h} = A\hat{z}_{t+h-1} + Bu_{t+h-1}, h = 1, 2, \cdots, H\}$. Thus the loss function is defined as:

$$\mathcal{L}_{koopman} = \frac{1}{H}\sum_{h=1}^{H}\gamma^h \left(\underbrace{\|\hat{z}_{t+h} - z_{t+h}\|^2}_{\mathcal{L}_{linear}} + 0.1 * \underbrace{\|\hat{x}_{t+h} - x_{t+h}\|^2}_{\mathcal{L}_{recon}} \right) \tag{6}$$

where $\hat{x}_{t:t+H} = P\hat{z}_{t:t+H}$, α is the weight of $\mathcal{L}_{recon}$ and γ is the discount factor. Here $\mathcal{L}_{linear}$ focuses on the linearization effect, while $\mathcal{L}_{recon}$ addresses the reconstruction effect. Since $z_t = [x_t, g'(x_t)]^T$, $\mathcal{L}_{linear}$ inherently implies $\mathcal{L}_{recon}$. Thus, slighty emphasizing reconsrtuction of original state by setting α to 0.1 shows better performance in practice. Then we can derive dynamics $\mathcal{T}^{(k)} \doteq (g^{(k)}_{n^{(k)}}, A^{(k)}_{n^{(k)}}, B^{(k)}_{n^{(k)}})$ for dataset $\mathcal{D}^{(k)}$, here $n^{(k)}$ represents the latent space dimension of iteration k.

3.3. Continual Learning and Lifting

The incremental Koopman algorithm begins with the initial dynamics $\mathcal{T}^{(0)}$ trained on the initial dataset $\mathcal{D}^{(0)}$. During the lifting phase, we use the increment data collector introduced in Section 3.1 to gather the incremental dataset $\mathcal{D}^{(1)}_{incre}$, which is then used to augment the initial dataset, forming $\mathcal{D}^{(1)} = \mathcal{D}^{(0)} \cup \mathcal{D}^{(1)}_{incre}$. Simultaneously, to accommodate the growing dataset, the latent space dimension is updated as $n^{(1)} = n^{(0)} + \Delta n$, where Δn is a hyperparameter chosen based on the trade-off between computational complexity, performance gain, and the requirement $m = \Omega(n \log(n))$ to be introduced in Section 3.4, where m denotes the dataset size.

In the learning phase, we utilize the loss introduced in Section 3.2 to obtain $\mathcal{T}^{(1)} = (g^{(1)}_{n^{(1)}}, A^{(1)}_{n^{(1)}}, B^{(1)}_{n^{(1)}})$. This process is repeated until the survival steps T_{sur} under MPC control with dynamics $\mathcal{T}^{(k)}$ cease to show improvement over $\mathcal{T}^{(k-1)}$. Check Algorithm 1 of our extended version of this paper Li et al. (2024) for details. While sampling all state-action pairs is unrealistic and unscalable in high-dimensional problems, our method balances sample complexity, approximation quality, and scalability. It leverages on-policy exploration to gradually minimize approximation errors within the

MPC controller's region of attraction, enabling reliable decisions and establishing robust linear latent dynamics. We validate our claims in Section 4.

3.4. Theoretical analysis

Incremental increases in the dimension of latent space and data size are central to our proposed algorithm. This section demonstrates that, under our incremental strategy, the learned Koopman operator matrix K converges to the true Koopman operator $\mathcal{K}$. For theoretical analysis, we primarily focus on autonomous systems as described in (1). For non-autonomous systems, the methodology extends by setting $s = [x; u]^\top$. The fundamental premise, as established in Korda and Mezić (2018a,b), is that the learned Koopman operator K is the L_2 projection of the true Koopman operator $\mathcal{K}$ onto the span of the n-dimensional embedding functions $\phi(s) = [\phi_1(s), \cdot, \phi_n(s)]$, where $\phi_i(s)$ is the one-dimensional embedding function for the i-th dimension of the latent space. Unlike the convergence analysis in Korda and Mezić (2018a), our theoretical framework introduces the convergence rate of our incremental Koopman algorithm. The detailed proof can be found in Appendix D of Li et al. (2024).

Theorem 1. *Under the assumptions of 1) data samples $s_1, \cdot, s_m$ are i.i.d distributed; 2) the latent state is bounded, $\|\phi(s)\| < \infty$; 3) the embedding functions $\phi_1, \cdots, \phi_n$ are orthogonal (independent). (a) The learned Koopman operator converges to the true Koopman Operator:*

$$\lim_{m=\Omega(nln(n)),n\to\infty} K \to \mathcal{K}. \tag{7}$$

(b) Assuming additional conditions: 4) The eigenvalues of $\mathcal{K}$ decay sufficiently fast, i.e., $|\lambda_i| \leq \frac{C}{i}$ for some constant $C > 0$. 5) The embedding functions $\phi(\cdot) = [\phi_1, \cdots, \phi_n]^\top$ correspond to the first n eigenfunctions of $\mathcal{K}$ associated with the largest eigenvalues in magnitude. Then the convergence rate of the linear approximation error is given by:

$$\textit{error} \leq \mathcal{O}(\sqrt{\frac{\ln(n)}{m}}) + \mathcal{O}\left(\frac{1}{\sqrt{n}}\right). \tag{8}$$

As indicated in Theorem 1, the proposed algorithm, by incrementally increasing the latent dimension n and data size m, progressively reduces the linear approximation error. According to Theorem 1(b), if the sample size meets $m = \Omega(n \ln(n))$, this reduction error follows the rate of $\mathcal{O}(n^{-\frac{1}{2}})$ as the dimension increases. In our experiments, we adhere to this guideline to ensure adequate sample generation. We note that the above assumptions are generally mild and hold in many practical scenarios. Further details on the theorem and underlying assumptions can be found in Appendix D of Li et al. (2024).

4. Experiments

In our experiments, we evaluate the proposed method by addressing the following questions: **Q1** Does the our learned Koopman Dynamics achive better state prediction performance compared to baselines? **Q2** Does our approach, combining learned Koopman dynamics with MPC control, achieve superior tracking performance relative to baselines? **Q3** Is the continual expansion of the dataset size in our proposed method effective? **Q4** Is the continual increase in the dimensions of the latent state beneficial?

4.1. Experiment Setup

Task settings. Our experiments use IsaacLab Mittal et al. (2023) as the simulation platform, the most advanced simulator offering comprehensive and flexible support for a wide range of robots and environments. Five different legged robots are included in our experiments: (i) **ANYmal-D:** A quadruped robot ($\mathcal{U} \subseteq \mathbb{R}^{12}$) designed by ANYbotics. (ii) **Unitree-A1:** A quadruped robot ($\mathcal{U} \subseteq \mathbb{R}^{12}$) designed by Unitree. (iii) **Unitree-Go2:** A quadruped robot ($\mathcal{U} \subseteq \mathbb{R}^{12}$) designed by Unitree. (iiiv) **Unitree-H1:** A Humanoid ($\mathcal{U} \subseteq \mathbb{R}^{19}$) designed by Unitree. (v) **Unitree-G1:** A Humanoid ($\mathcal{U} \subseteq \mathbb{R}^{23}$) designed by Unitree. Furthermore, two different types of terrain are considered, including (i) **Flat:** flat terrain. (ii) **Rough:** rough terrain with random surface irregularities, where the height follows a uniform distribution between 0.005 and 0.025, and the minimum height variation is 0.005 (in m). All test suites are based on the task **Walk**: tracking a velocity command with uniformly sampled heading direction, x-axis linear velocity, and y-axis linear velocity. Considering these settings, we design 7 test suites with 5 types of legged robots and 2 types of terrain. We name these test suites as `{Terrain}-{Make}-{Model}`. Please check Appendix E of Li et al. (2024) for experiment details.

Comparison Group. We compare Incremental Koopman algorithm with state-of-the-art Koopman-based algorithms (i) Deep KoopmanU with Control (DKUC, Shi and Meng (2022)) algorithm, (ii) Deep Koopman Affine with control (DKAC, Shi and Meng (2022)) algorithm, (iii) Neural Network Dynamics Model (NNDM, Nagabandi et al. (2017); Liu et al. (2023)) with Nonlinear MPC (iiiv) Deep Koopman Reinforcement Learning (DKRL, Song et al. (2021)). It worth noting that all methods are trained seperately on 7 test suites. And for all experiments, we take the best algorithm-specific parameters mentioned in the original paper and keep the common parameters the same.

Metrics. We introduce the following metrics to evaluate the tracking ability of our algorithm: (i) **Prediction-based:** we leverage k-step prediction error $E_{pre}(k) \doteq \frac{1}{kn'} \sum_{t=1}^{k} \|x_t - x_t^*\|_1$ $(x_0 = x_0^*)$ generally used for measuring prediction ability of given dynamics. (ii) **Pose-based:** To evaluate tracking accuracy, we evaluate Joint-relative mean per-joint position error(E_{JrPE}), Joint-relative mean per-joint velocity error (E_{JrVE}), Joint-relative mean per-joint acceleration error (E_{JrAE}), Root mean position error (E_{RPE}), Root mean orientation error (E_{ROE}), Root mean linear velocity error (E_{RLVE}) and Root mean angular velocity error (E_{RLAE}). (iii) **Physics-based:** Considering the easy-to-fail feature of legged robots, for the same reference repository $\mathcal{R}$, we evaluate the average survival simulation time step T_{Sur} (set 200 as upper bound, simulation frequency is 50Hz) for each algorithm, deeming tracking unsuccessful when E_{JrPE} is larger than ϵ_{fail}. Check Appendix E.3 and E.4 of Li et al. (2024) for details.

State. The system state $x_t \doteq \begin{cases} (j_t, \dot{j}_t, p_t^z, \dot{p}_t, \dot{r}_t), \textit{ for humanoid} \\ (j_t, \dot{j}_t, p_t^z, \dot{p}_t, r_t, \dot{r}_t), \textit{ for quadruped robot} \end{cases}$ of dynamics and controller includes the joint position j_t and joint velocity $\dot{j}_t$ in local coordinates, root height p_t^z, 3D root linear velocity p_t, 4D root orientation r_t (optional for humanoid and represented as a quaternion), and 3D root angular velocity $\dot{r}_t$ in world coordinates. All of the aforementioned quantities are normalized to follow a standard Gaussian distribution, $\mathcal{N}(0, 1)$, in order to eliminate inconsistencies in the scale of different physical quantities.

Control Input and Low-level Controller. We use a proportional-derivative (PD) controller at each joint of the legged robots as the low-level controller. Thus, the control input u_t determined by the MPC serves as the joint target, denoted as $j_t^d = u_t$. The torque applied to each joint is given by

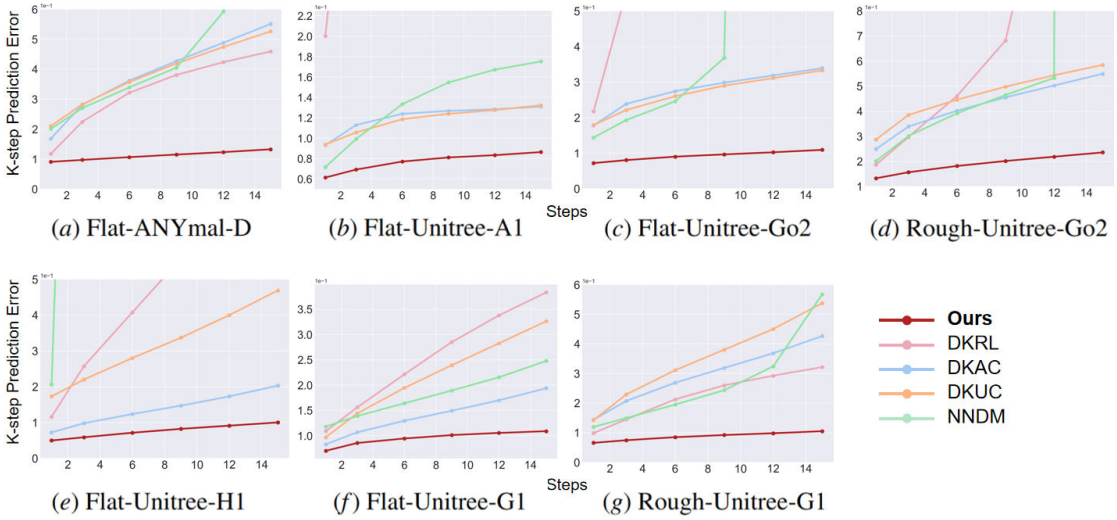


Figure 2: Comparison results of k-step prediction error in our test suites.

$\tau_t = k^p \circ (j_t^d - j_t) - k^d \circ \dot{j}_t$, where k^p and k^d are the proportional and derivative gains, respectively. The low-level controller operates at a frequency of 200 Hz, with a decimation factor of 4.

4.2. K-step Prediction Error

Specifically, we set k to $[1, 3, 6, 9, 12, 15]$ and compute $E_{pre}(k)$ for each algorithm, subsequently plotting the results in Figure 2. The results highlight our algorithm's exceptional ability to maintain low prediction errors, even over long horizons in complex, nonlinear tasks—challenging for other algorithms in our comparison. Figures 3(c) to 3(e) show that NNDM and DKRL struggle with the accumulation of explosive errors as horizon length k increases, while our algorithm remains stable, with negligible error growth. Though DKAC and DKUC manage a stable rate of loss increase, they exhibit relatively high prediction errors due to their limited subspace modeling compared to our incremental method. These results address **Q1**.

4.3. Tracking Performance

Figure 3: Visualization of tracking performance on Flat-Unitree-G1

To highlight the advantage of our method in tracking tasks, we track 3000 references in $\mathcal{R}$ over 200 steps in parallel, presenting the results in Table 1 with average tracking metrics from Section 4.1. Further details are provided in Appendix E.5 of Li et al. (2024).

Compared to other algorithms, our approach shows (i) high T_{Sur} values nearing the upper bound of 200, (ii) precise, continuous joint-level tracking, and (iii) sustained global tracking with balance

Synthesised Tracking Metrics								
Algorithm	Joint-relative ↓			Root-relative ↓				Survival ↑
	E_{JrPE}	E_{JrVE}	E_{JrAE}	E_{RPE}	E_{ROE}	E_{RLVE}	E_{RAVE}	T_{Sur}
Ours with MPC	**0.0348**	**0.6499**	**43.1465**	**0.1231**	**0.0668**	**0.1216**	**0.3289**	**188.4514**
DKRL with MPC	0.0823	1.1251	68.9520	0.2978	0.1561	0.2089	0.5634	116.9540
DKAC with MPC	0.1816	2.0694	117.5515	0.3955	0.2749	0.2888	0.9143	25.0254
DKUC with MPC	0.1576	1.0828	50.5745	0.2934	0.1989	0.2252	0.5559	82.4620
NNDM with NMPC	0.1439	2.2020	127.4524	0.4334	0.2506	0.2996	0.8536	35.4709

Table 1: The average tracking metrics evaluated for each algorithm across all 7 test suites.

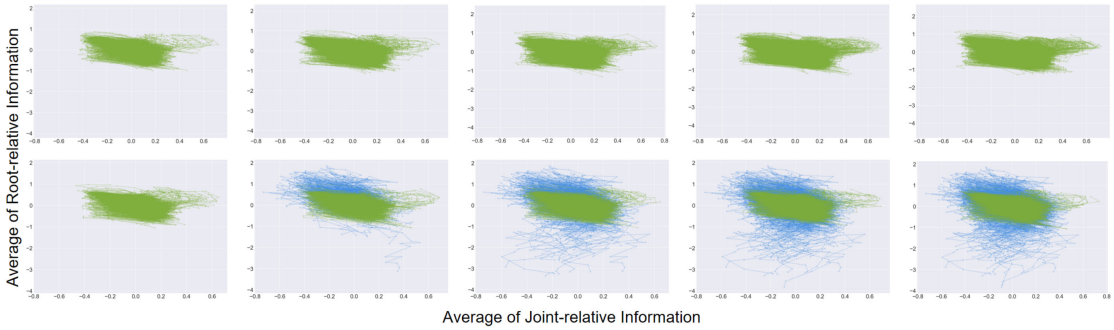


Figure 4: Visualization of the data distribution by plotting the means of the joint-relative and root-relative states. The first row illustrates the dataset expansion achieved using an RL policy, while the second row shows the results from applying our algorithm for dataset expansion.

and stability. All measured metrics are significantly lower for competing algorithms, which often fail to track correctly and fall after only a few attempts. In contrast, our algorithm delivers efficient, accurate tracking with a steady gait, demonstrating its ability to infer corner cases during tracking, thanks to the continual learning and lifting process. Meanwhile, others struggle to achieve the correct poses and gait when faced with falling. Notably, the 200-step upper bound for T_{Sur} suggests that many test cases remain robust enough for even longer tracking durations.

Although DKRL and DKUC achieve relatively high T_{Sur}, their E_{JrPE} and E_{RPE} are 2 to 5 times higher than ours, indicating that their metrics include brief failure episodes. In contrast, our algorithm maintains robust tracking throughout, showcasing superior steady tracking with minimal error. DKAC and NNDM, however, suffer from short inference capabilities and sensitivity to the dynamics' corner cases, making them prone to failure—especially in situations where recovery is impossible for legged robots. As shown in Figure 3, unrecoverable failures end the tracking process, answering **Q2**.

4.4. Ablation on Dataset Increment

For the ablation study on dataset increment, we select Flat-Unitree-Go2 and Flat-Unitree-G1 to examine the impact of dataset increment techniques. As shown in Table 2 and fig. 5, the k-step prediction error rises sharply with increasing steps, highlighting limited inferencing capability when dynamics encounter corner cases due to insufficient latent subspace modeling. Furthermore, E_{JrPE} is nearly seven times higher than the original method, indicating frequent failure scenarios. The low survival steps T_{Sur} further corroborate this observation. In contrast, the original algorithm maintains exceptional tracking performance and loss control, demonstrating the effectiveness of

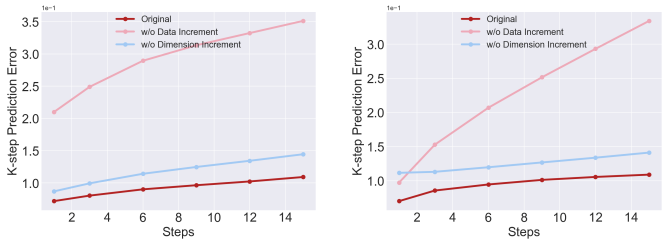
Average Tracking Metrics of Flat-Unitree-Go2 and Flat-Unitree-G1								
Algorithm	Joint-relative ↓			Root-relative ↓				Survival ↑
	E_{JrPE}	E_{JrVE}	E_{JrAE}	E_{RPE}	E_{ROE}	E_{RLVE}	E_{RAVE}	T_{Sur}
Original	**0.0246**	**0.5954**	**42.1403**	**0.0673**	**0.0245**	**0.0650**	**0.2060**	**196.6190**
w/o Data.I.	0.2061	1.5251	65.5157	0.3452	0.2449	0.2740	0.6303	53.0540
w/o Dim.I.	0.1189	1.1743	63.6319	0.2526	0.1936	0.2178	0.5782	100.9350

Table 2: The average tracking metrics evaluated for the ablation studies. 'w/o Data.I.' and 'w/o Dim.I.' indicate the performance of the original method without data increment and dimension increment techniques, respectively

dataset enlargement. Additionally, Figure 4 visualizes the data enlargement process, showing that RL policy-based expansion is ineffective due to repetitive data, while our algorithm achieves effective dataset growth. This underscores the importance of robust subspace modeling and accurate inferencing, addressing **Q3**.

4.5. Ablation on Dimension Increment

To conduct the ablation study on dimension increment, we select the same test suites mentioned in Section 4.4. As shown in Figure 5, the k-step prediction error remains steady and relatively low, similar to the original methods, benefiting from the continual increment of the dataset to model the subspace. However, as indicated in Table 2, although the algorithm without dimension increment outperforms the algorithm without data increment, its tracking performance still falls significantly short of the original method. This suggests that a low latent space dimension fails to model robust dynamics strong enough to resist unseen noise during simulation. Furthermore, the lack of inferencing capability highlights its shortcomings in linearization and generalization. Thus, the dimension increment technique emphasizes the importance of modeling robust and general sub-dynamics, addressing **Q4**.



(*a*) Flat-Unitree-Go2 (*b*) Flat-Unitree-G1

Figure 5: K-step prediction error comparison of ablation on dimension and dataset increment.

5. Conclusion and Future Works

By progressively expanding the dataset and latent subspace, our method ensures convergence of linearization errors and enables accurate approximations of the system dynamics. Experimental results show that the approach achieves high control performance with simple MPC controllers on various tasks after just a few iterations. This work represents the first successful application of linearized Koopman dynamics to locomotion control of legged robots, offering a scalable and model-based control solution. At the same time, the incremental Koopman algorithm may encounter an explosion in latent dimensionality when the subspace expands rapidly. In the future, tele-operation or retargeted human data will be tested to implement our algorithm in real-world scenarios.

Acknowledgments

This research is supported by the National Science Foundation (NSF) under Grant No. 2144489.

References

Zipeng Fu, Qingqing Zhao, Qi Wu, Gordon Wetzstein, and Chelsea Finn. Humanplus: Humanoid shadowing and imitation from humans, 2024. URL https://arxiv.org/abs/2406.10454.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. PMLR, 2018.

Yunhai Han, Mandy Xie, Ye Zhao, and Harish Ravichandar. On the utility of koopman operator theory in learning dexterous manipulation skills. In *Conference on Robot Learning*, pages 106–126. PMLR, 2023.

Tairan He, Zhengyi Luo, Wenli Xiao, Chong Zhang, Kris Kitani, Changliu Liu, and Guanya Shi. Learning human-to-humanoid real-time whole-body teleoperation. *arXiv preprint arXiv:2403.04436*, 2024.

Hamed Kazemi, Vahid Johari Majd, and Majid M. Moghaddam. Modeling and robust backstepping control of an underactuated quadruped robot in bounding motion. *Robotica*, 31(3):423–439, 2013. doi: 10.1017/S0263574712000458.

Jeonghwan Kim, Yunhai Han, Harish Ravichandar, and Sehoon Ha. Learning koopman dynamics for safe legged locomotion with reinforcement learning-based controller. *arXiv preprint arXiv:2409.14736*, 2024.

Milan Korda and Igor Mezić. On convergence of extended dynamic mode decomposition to the koopman operator. *Journal of Nonlinear Science*, 28:687–710, 2018a.

Milan Korda and Igor Mezić. Linear predictors for nonlinear dynamical systems: Koopman operator meets model predictive control. *Automatica*, 93:149–160, 2018b.

Alexander Krolicki, Dakota Rufino, Andrew Zheng, Sriram S.K.S Narayanan, Jackson Erb, and Umesh Vaidya. Modeling quadruped leg dynamics on deformable terrains using data-driven koopman operators. *IFAC-PapersOnLine*, 55(37):420–425, 2022. ISSN 2405-8963. doi: https://doi.org/10.1016/j.ifacol.2022.11.219. URL https://www.sciencedirect.com/science/article/pii/S2405896322028622. 2nd Modeling, Estimation and Control Conference MECC 2022.

Joonho Lee, Jemin Hwangbo, Lorenz Wellhausen, Vladlen Koltun, and Marco Hutter. Learning quadrupedal locomotion over challenging terrain. *Science robotics*, 5(47):eabc5986, 2020.

Feihan Li, Abulikemu Abuduweili, Yifan Sun, Rui Chen, Weiye Zhao, and Changliu Liu. Continual learning and lifting of koopman dynamics for linear control of legged robots, 2024. URL https://arxiv.org/abs/2411.14321.

Ziang Liu, Genggeng Zhou, Jeff He, Tobia Marcucci, Li Fei-Fei, Jiajun Wu, and Yunzhu Li. Model-based control with sparse neural dynamics, 2023. URL https://arxiv.org/abs/2312.12791.

Zhengyi Luo, Jinkun Cao, Kris Kitani, Weipeng Xu, et al. Perpetual humanoid control for real-time simulated avatars. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10895–10904, 2023.

Xubo Lyu, Hanyang Hu, Seth Siriya, Ye Pu, and Mo Chen. Task-oriented koopman-based control with contrastive encoder. In *Conference on Robot Learning*, pages 93–105. PMLR, 2023.

Giorgos Mamakoukas, Maria L Castano, Xiaobo Tan, and Todd D Murphey. Derivative-based koopman operators for real-time control of robotic systems. *IEEE Transactions on Robotics*, 37 (6):2173–2192, 2021.

Mayank Mittal, Calvin Yu, Qinxi Yu, Jingzhou Liu, Nikita Rudin, David Hoeller, Jia Lin Yuan, Ritvik Singh, Yunrong Guo, Hammad Mazhar, Ajay Mandlekar, Buck Babich, Gavriel State, Marco Hutter, and Animesh Garg. Orbit: A unified simulation framework for interactive robot learning environments. *IEEE Robotics and Automation Letters*, 8(6):3740–3747, 2023. doi: 10.1109/LRA.2023.3270034.

Anusha Nagabandi, Gregory Kahn, Ronald S. Fearing, and Sergey Levine. Neural network dynamics for model-based deep reinforcement learning with model-free fine-tuning, 2017. URL https://arxiv.org/abs/1708.02596.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017. URL https://arxiv.org/abs/1707.06347.

Haojie Shi and Max Q-H Meng. Deep koopman operator with control for nonlinear systems. *IEEE Robotics and Automation Letters*, 7(3):7700–7707, 2022.

Lu Shi, Masih Haseli, Giorgos Mamakoukas, Daniel Bruder, Ian Abraham, Todd Murphey, Jorge Cortes, and Konstantinos Karydis. Koopman operators in robot learning, 2024. URL https://arxiv.org/abs/2408.04200.

Lixing Song, Junheng Wang, and Junhong Xu. A data-efficient reinforcement learning method based on local koopman operators. In *2021 20th IEEE International Conference on Machine Learning and Applications (ICMLA)*, pages 515–520, 2021. doi: 10.1109/ICMLA52953.2021.00086.

Masaki Murooka Sotaro Katayama and Yuichi Tazaki. Model predictive control of legged and humanoid robots: models and algorithms. *Advanced Robotics*, 37(5):298–315, 2023. doi: 10.1080/01691864.2023.2168134. URL https://doi.org/10.1080/01691864.2023.2168134.

Pierre-Brice Wieber, Russ Tedrake, and Scott Kuindersma. Modeling and control of legged robots. In *Springer Handbook of Robotics, 2nd Ed.*, 2016. URL https://api.semanticscholar.org/CorpusID:6790710.

Zeji Yi, Chaoyi Pan, Guanqi He, Guannan Qu, and Guanya Shi. Covo-mpc: Theoretical analysis of sampling-based mpc and optimal covariance design, 2024. URL https://arxiv.org/abs/2401.07369.

Zixin Zhang, John Z. Zhang, Shuo Yang, and Zachary Manchester. Robots with attitude: Singularity-free quaternion-based model-predictive control for agile legged robots, 2024. URL https://arxiv.org/abs/2409.09940.

Weiye Zhao, Feihan Li, Yifan Sun, Rui Chen, Tianhao Wei, and Changliu Liu. Absolute policy optimization: Enhancing lower probability bound of performance with high confidence. In *Forty-first International Conference on Machine Learning*.

James Zhu, J. Joe Payne, and Aaron M. Johnson. Convergent ilqr for safe trajectory planning and control of legged robots, 2024. URL https://arxiv.org/abs/2304.00346.