MORALS: Analysis of High-Dimensional Robot Controllers via Topological Tools in a Latent Space

Ewerton R. Vieira[*,2,3], Aravind Sivaramakrishnan[*,1], Sumanth Tangirala[1], Edgar Granados[1], Konstantin Mischaikow[2], and Kostas E. Bekris[1]

***Abstract*— Estimating the region of attraction (RoA) for a robot controller is essential for safe application and controller composition. Many existing methods require a closed-form expression that limit applicability to data-driven controllers. Methods that operate only over trajectory rollouts tend to be data-hungry. In prior work, we have demonstrated that topological tools based on *Morse Graphs* (directed acyclic graphs that combinatorially represent the underlying nonlinear dynamics) offer data-efficient RoA estimation without needing an analytical model. They struggle, however, with high-dimensional systems as they operate over a state-space discretization. This paper presents *Mo*rse Graph-aided discovery of *R*egions of *A*ttraction in a learned *L*atent *S*pace (MORALS)[**]. The approach combines auto-encoding neural networks with Morse Graphs. MORALS shows promising predictive capabilities in estimating attractors and their RoAs for data-driven controllers operating over high-dimensional systems, including a 67-dim humanoid robot and a 96-dim 3-fingered manipulator. It first projects the dynamics of the controlled system into a learned latent space. Then, it constructs a reduced form of Morse Graphs representing the bistability of the underlying dynamics, i.e., detecting when the controller results in a desired versus an undesired behavior. The evaluation on high-dimensional robotic datasets indicates data efficiency in RoA estimation.**

I. Introduction

Given a controller for a robotic system, it is desirable to estimate its region of attraction (RoA), i.e., a subset of the system's state space, such that all trajectories starting inside this set converge to an equilibrium [1]. RoA estimation helps understand the conditions under which the controller can be safely applied to solve a task. It can also be used for controller composition where the final RoA is larger than the RoA of each component controller [2].

The authors introduced in prior work topological tools based on *Morse graphs* [3]. Morse Graphs provide a finite, combinatorial representation of the state space given access to a discrete-time representation of the dynamics. They correspond to directed acyclic graphs that provide a rigorous description of attractors and RoAs at different levels of resolution. They were introduced as data-efficient and more accurate alternatives to estimate the RoAs of general robot controllers, including data-driven ones. For systems with unknown dynamics, they can be combined with surrogate modeling to identify the RoA for a goal set and describe the global dynamic behavior of a controller [4].

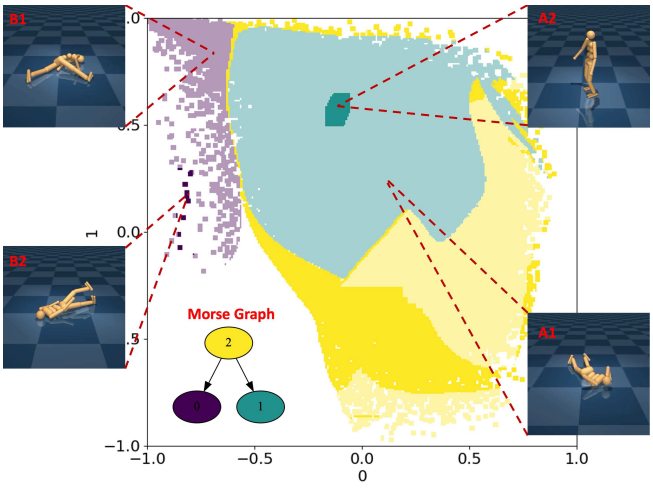

Fig. 1: The 67-dim. state space of a bipedal humanoid robot controlled by a Soft Actor-Critic (SAC) controller is encoded to a 2-dim. learned latent space by MORALS, which then discovers two attractors and their corresponding Regions of Attraction (RoAs) in the latent space. Encoded final states **A2**, **B2** are mapped to a desired (dark green) and undesired attractor (dark purple) respectively. Encoded initial states **A1**, **B1** lie respectively in the RoAs (light green and light purple) of the desired and undesired attractors. The yellow region contains the separatrix (undecidable region), indicating initial states may go to **A2** (node 2 → node 1) or **B2** (node 2 → node 0). Best viewed in color.

Morse Graphs rely only on point-wise access to short trajectories from each cell of a state space discretization. Their accuracy depends significantly on the size of the discretization of the system's state space. Thus, applying them directly to high-dim. robotic systems, such as bipedal robots (Fig 1), is computationally expensive or even infeasible. In practice, however, the effect of a controller on a robotic system, even a complex and high-dim. one is to restrict the dynamics to a lower-dim. manifold. For example, simplified models of either stiff [5] or compliant [6] inverted pendula can result in controllers for a high-dim. bipedal robot. It may thus be possible to derive meaningful conclusions about the dynamics in the robot's original high-dim. state space by first identifying a lower-dim manifold given example trajectory data and studying the dynamics of that manifold.

This work proposes **Mo**rse Graph-aided discovery of **R**egions of **A**ttraction in a learned **L**atent **S**pace (MORALS), which uses an autoencoding neural network as a lower-dim. surrogate model of the underlying controlled dynamics trained on robot trajectories. The network encodes the

*The first two authors contributed equally to this paper.

**Code: https://github.com/Ewerton-Vieira/MORALS

[1] Dept. of Computer Science, [2] Dept. of Mathematics and [3] DIMACS, Rutgers, NJ, USA. {er691,as2578, kb572}@rutgers.edu.

This work is partly supported by NSF HDR TRIPODS award 1934924. EV and KM are partially supported by Air Force Office of Scientific Research under award numbers FA9550-23-1-0011 and FA9550-23-1-0400.

system's high-dim. state space to a lower-dim *latent* one so that the latent dynamics locally approximate the system's behavior in the original space. Then, a discretization of the low-dim. space is used to compute a combinatorial representation of the latent dynamics in the form of a Morse Graph, i.e., a directed acyclic graph that provides a rigorous description of the attractors and RoAs at different levels of resolution. Finally, the Morse Graph is reduced into a simpler graph representing the bistability in the dynamics, i.e., identifying desired versus undesired behaviors of the controller as shown in Fig 1. The experimental evaluation considers a combination of analytical and physically simulated benchmarks, including for a 67-dim. humanoid, as well as real-world robotic datasets - for a 96-dim 3-fingered manipulator. Due to both dimensionality and non-availability of the ground-truth model, it is infeasible to directly apply the prior Morse Graph framework. Nevertheless, MORALS achieves promising accuracy with significantly fewer data requirements in estimating the desired RoA. *To the best of the authors' knowledge, no competing methods currently can perform this analysis for black-box controllers of such high-dim. systems given only trajectory data.*

II. Related work

Estimating the RoA of a control system is a hard problem [7]. Multiple methods [8] rely on an analytical expression and use linear matrix inequalities [9], [10] or sum-of-squares solvers [2], [11], [12]. Traditional Lyapunov-based methods are applicable [11], [13] but require an analytical model. Even data-driven variants tend to require access to point-wise evaluation of the dynamical model [14]. Conservative approximation methods try to estimate the largest possible set within the true RoA [15]–[18]. Gaussian Processes can provide Lyapunov-like functions [19]. These methods typically suffer from high data requirements and estimation of a single attractor.

Reachability analysis [1] and control barrier functions [20] are popular alternatives. Reachability analysis can approximate the RoA of dynamical walkers [21] and together with learning can maintain system safety over a given horizon [22]. Using Gaussian Processes, a barrier function can be learned to obtain safe policies [23], or identify areas needing exploration for safe set expansion [24]. Controllers can also be jointly trained alongside neural Lyapunov functions [25]. Prior knowledge of an attractor is a common requirement for these approaches. In contrast, the proposed approach can discover multiple if they exist. Alternative methods also lack explainability, while MORALS provides a graphical description of the dynamics.

Unsupervised representation learning helps extract a latent representation of a robot's state space and enforces the dynamics in this learned space. Such efforts [26]–[28] tend to focus on locally valid dynamics and do not study the global dynamics. Latent Sampling-based Motion Planning (L-SBMP) [29] enforces the latent dynamics via a reachability Grammian. Similarly, Learning To Correspond Dynamical Systems (L2CDS) [30] learns correspondences between pairs of dynamical systems through a shared latent dynamical system. MORALS uses an autoencoding architecture similar to L-SBMP and L2CDS. Unlike L-SBMP, it does not learn the latent dynamics for the system but focuses on the controller studied. Unlike L2CDS, MORALS does not require assumptions about the latent dynamical model and learns the latent manifold directly from data.

III. Preliminaries

This work aims to provide a data-efficient framework for the analysis of the global dynamics of robot controllers based on combinatorial dynamics and order theory [3], [31]–[33]. Consider a non-linear, continuous-time control system:

$$\dot{x} = f(x, u) \tag{1}$$

where $x(t) \in X \subseteq \mathbb{R}^N$ is the state at time t, X is a compact set, $u : X \mapsto U \subseteq \mathbb{R}^M$ is a Lipschitz-continuous control as defined by a deterministic control policy $u(x)$, and $f : X \times \mathbb{U} \mapsto \mathbb{R}^M$ is a Lipschitz-continuous function. Neither $f(\cdot)$ nor $u(x)$ are necessarily known analytically. For instance, $u(x)$ can be a function learned via a neural network.

A trajectory (or an *orbit*) of length $\tau > 0$ is defined as a sequence of states obtained by integrating Eq. 1 forward in time. Let the *image* $\phi_\tau : X \to X$ denote the function obtained by mapping every initial state $x_0 \in X$ to the end state of a trajectory of length τ beginning at x_0. A set A is an attractor if there exists a neighborhood N of A such that

$$\mathcal{A} = \omega(N) := \bigcap_{n \in \mathbb{Z}^+} \mathrm{cl}\left(\bigcup_{k=n}^{\infty} \phi_\tau^k(N)\right)$$

where ϕ_τ^k is the composition $\phi_\tau \circ \cdots \circ \phi_\tau$ (k times) and cl is topological closure. For instance, attractors can be fixed points, such as a desired goal that the control policy manages to bring the system to; or limit cycles, such as a periodic behavior of the system. A *Region of Attraction* (RoA) of an attractor A is a neighborhood of A and a subset of $\mathcal{B}$ the *basin of attraction* A, where $\mathcal{B}$ is the largest set of points whose forward orbits converge to A, more specifically, the maximal set $\mathcal{B}$ that satisfies $A = \omega(\mathcal{B})$. Given that f and u are Lipschitz-continuous, ϕ_τ is too; also any RoA of Eq. (1) is an RoA under ϕ_τ. Thus, it is sufficient to analyze the dynamics according to ϕ_τ to study Eq. (1), even if it is not accessible and computable.

As a pedagogical example, consider the N-dim. *bistable* system in Fig. 2: Given a state $x = [x_1, \cdots, x_N] \in X = X_1 \times \prod_{i=2}^N X_i = [-3, 3] \times [-2, 2]^{N-1}$, its image is given by $\phi_\tau(x) = \arctan 4x_1 \times \prod_{i=2}^N x_i/2$. Obtaining Morse graphs and RoAs involves a four-step procedure:

1. State space decomposition and outer approximation of ϕ_τ. The function ϕ_τ is approximated by decomposing X into a collection of regions $\mathcal{X}$. For instance, via a grid. Fig. 2 (left) shows the intervals $[-3, 3]$ and $[-2, 2]$ decomposed into sub-intervals a to e and f_i to h_i respectively. A grid will be the Cartesian product $\{a, b, c, d, e\} \times \prod_{i=2}^N \{f_i, g_i, h_i\}$. Given a region $\xi \in \mathcal{X}$ (i.e., a cell in the grid), the system is forward propagated for multiple initial states within ξ for time τ to identify regions reachable from ξ. Consider the sub-interval $b \times \prod g_i$. The lines from the boundary of $b \times \prod g_i$ depict

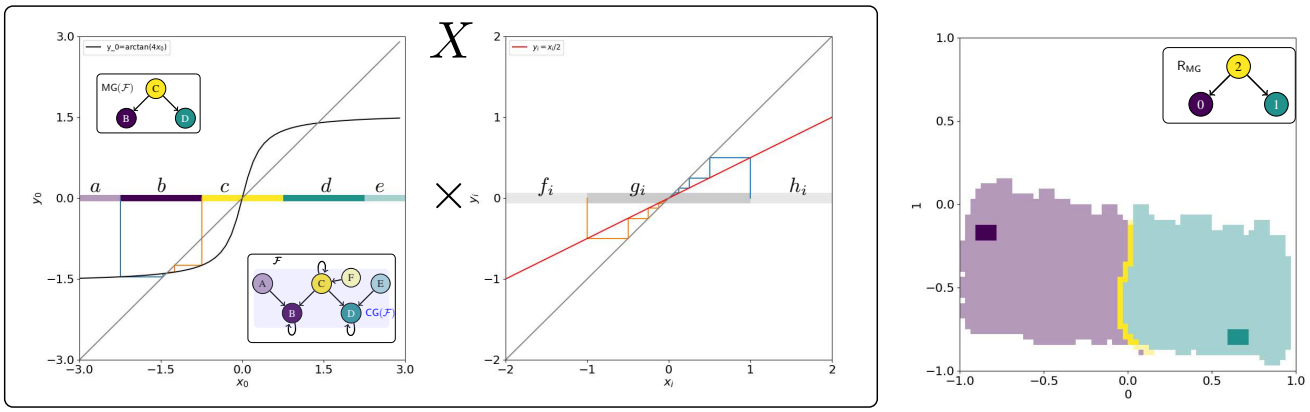

Fig. 2: Example of N-dim. bistability (left and middle) and the learned dynamics on the 2-dim. encoded space (right). (left & middle) The state space is $X = X_1 \times \prod_{i=2}^{N} X_i$ where $X_i = [-3, 3] \times [-2, 2]^{N-1}$ and the dynamics $\phi : X \to X$ is given by $\phi_1(x) = \arctan(4x)$ plotted in black and $\phi_i(x) = x/2$ plotted in red, where $i = 2, \ldots, N$. The domains $X_1 = [-3, 3]$ and $X_i = [-2, 2]$ are decomposed into intervals a through e and f_i, g_i, h_i, respectively. Forward propagation of $B = b \times \prod g_i$ is depicted by the lines from the boundary of b and g_i's. $\mathcal{F}$ is a directed graph capturing reachable vertices from other vertices, (regions in $\{a, b, c, d, e\} \times \prod_{i=2}^{N} \{f_i, g_i, h_i\}$). Strongly connected components of $\mathcal{F}$ result in $\mathsf{CG}(\mathcal{F})$. Finally, the Morse Graph $\mathsf{MG}(\mathcal{F})$ (nodes B, C, D) contains the attractors and expresses their RoAs. (right) The N-dim. bistability dynamics are encoded into a 2-dim. latent space represented by a bistable Morse Graph R_{MG}.

the forward propagation of the dynamics. This cell maps to itself given the underlying dynamics.

2. Constructing combinatorial representation $\mathcal{F}$ of the dynamics. The directed graph representation denoted as $\mathcal{F}$ stores regions $\xi \in \mathcal{X}$ as vertices. Edges from ξ point to regions reachable from ξ. In Fig. 2, the graph $\mathcal{F}$ contains the following nodes/cells: $B = \{b\} \times \prod_{i=2}^{N} \{g_i\}$, $C = \{c\} \times \prod_{i=2}^{N} \{g_i\}$, $D = \{d\} \times \prod_{i=2}^{N} \{g_i\}$, $A = \{a, b\} \times \prod_{i=2}^{N} \{f_i, g_i, h_i\} \backslash B$, $E = \{d, e\} \times \prod_{i=2}^{N} \{f_i, g_i, h_i\} \backslash D$ and $F = \{c\} \times \prod_{i=2}^{N} \{f_i, g_i, h_i\} \backslash C$. The edges (C, C) and (C, B) express that C maps both to itself and to B.

3. Compute the Condensation Graph $\mathsf{CG}(\mathcal{F})$. Collapsing all nodes that are part of strongly connected components (SCCs) in $\mathcal{F}$ into a single node gives rise to the condensation graph $\mathrm{CG}(\mathcal{F})$. Edges on $\mathrm{CG}(\mathcal{F})$ reflect reachability due to the dynamics given a topological sort on $\mathcal{F}$. In Fig. 2(left), $\mathrm{CG}(\mathcal{F})$ is the subgraph with nodes A to F and all non-self edges, i.e., $\mathrm{CG}(\mathcal{F})$ has no cycles. Since $\mathrm{CG}(\mathcal{F})$ is a directed acyclic graph, it is a *partially ordered set*, i.e., a *poset*.

4. Compute the *Morse Graph* MG to detect attractors and RoAs. A *recurrent set* is a SCC of $\mathsf{CG}(\mathcal{F})$ that contains at least one edge. Then, the *Morse graph* $\mathrm{MG}(\mathcal{F})$ of $\mathcal{F}$ is the subgraph of recurrent sets of $\mathrm{CG}(\mathcal{F})$. In Fig. 2(left), $\mathrm{MG}(\mathcal{F})$ contains nodes B, C and D and the edges between them. $\mathrm{MG}(\mathcal{F})$ captures both recurrent and non-recurrent dynamics. Recurrent sets are vertices, and the minimal vertices contain attractors of interest. Edges signify reachability between these sets. In Fig. 2, there are 2 attractors: B and D. Cells in A are in the RoA of B, and cells in E are in the RoA of D. From cells in F and C, the system can end up in either B or D, characterizing a *bistability*.

As a summary of prior results with the Morse Graph approach, consider a pendulum governed by $ml^2\ddot{\theta} = mGl \sin\theta - \beta\dot{\theta} + u$, given mass m, gravity G, pole length l and friction coefficient β. It is controlled by a Linear Quadratic Regulator (LQR) to stand upright. Table I reports the RoA estimate accuracy (as a ratio over the volume of the true RoA) for the above approach (MorseGraph) and alternatives in the literature. Data efficiency is measured using the total propagation steps required for the RoA estimate. Both L-LQR and L-SOS are analytical methods that use a linearized unconstrained form of the dynamics [34] to obtain a Lyapunov function (LF). Lyapunov Neural Network (L-NN) [18] is a machine learning tool for estimating RoAs.

Metric	L-LQR	L-SOS	L-NN	MorseGraph
Accuracy	70%	3%	98%	97%
Prop. steps	–	–	667.1M	120K

TABLE I: Accuracy and data efficiency of desired RoA estimate for Pend (LQR) using different methods. L-LQR and L-SOS [3] require the analytical form of the dynamics.

The above methods either require access to an analytical model (L-LQR, L-SOS) or dynamics propagation from a dense set of initial states (L-NN). This may not be possible for complex and high-dim. robots or data-driven controllers that do not admit a closed-form expression. As the dimension of the underlying system increases, it becomes challenging to apply the MorseGraph approach, even though it requires $6000\times$ fewer data points than L-NN for a comparable RoA estimate. This is due to the exponential increase in the number of discretization elements. The proposed framework deals with this issue via unsupervised representation learning.

IV. Proposed Method

Given a high N-dim. state x, an *encoder* $h_{\text{enc}} : X \mapsto Z \subseteq \mathbb{R}^D (D < N)$ encodes x to a lower D-dim. latent state $z = h_{\text{enc}}(x)$. A *decoder* performs the inverse mapping $x = h_{\text{dec}}(z)$. A *latent dynamics* function, $h_{\text{dyn}} : Z \mapsto Z$, expresses the dynamics of the latent space. The proposed approach, MORALS, incorporates an autoencoding network consisting of h_{enc}, h_{dec}, and h_{dyn}, that can be trained from trajectory rollouts of the underlying system. The architecture used is shown in Fig. 3 (left). The trained networks are used to build a combinatorial representation of the dynamics in the learned latent space Z, which can be used to understand the RoAs

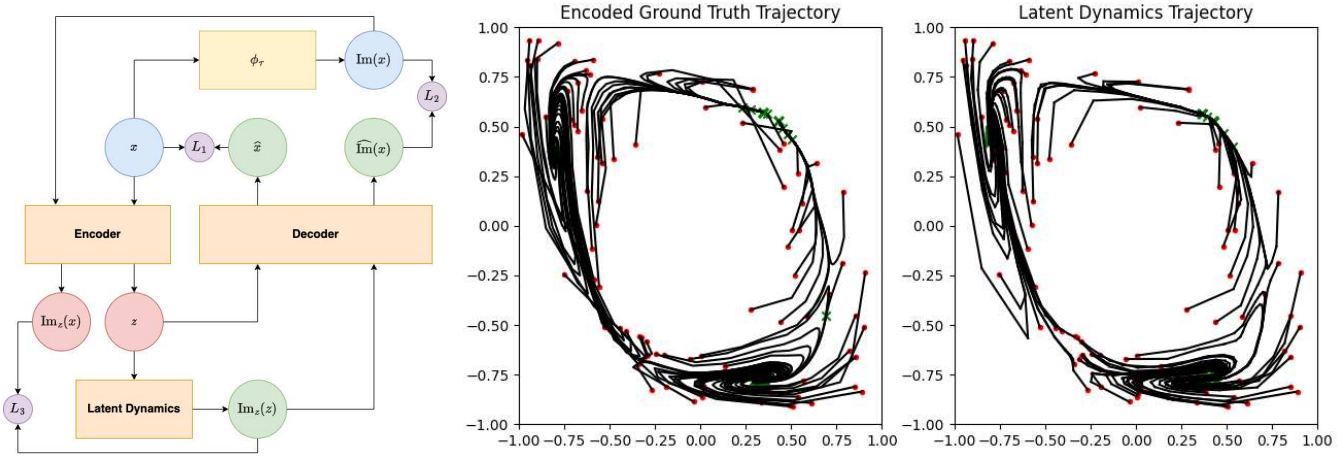

Fig. 3: (Left) The autoencoding neural network and loss functions used for training the encoder h_{enc}, decoder h_{dec}, and the latent dynamics h_{dyn}. (Middle/Right) Visualizing the learned latent dynamics for a 4-dim. version of a pendulum $(x, \dot{x}, y, \dot{y})$ controlled by LQR. (Middle) the ground-truth trajectory for the same initial conditions (circles). (Right) iteratively calling h_{dyn} for a fixed number of timesteps. Both plots capture the 3 true attractors ($\times$). The right plot does not contain regions where the trajectories move from one RoA to another.

of the system in the original state space X. The method is divided into the following steps:

1. Collect data from the system.
2. Train the neural networks $h_{\text{enc}}, h_{\text{dec}}$ and h_{dyn}.
3. Compute the Morse Graph MG of the lower dim. system with state space Z, whose initial condition is $z_0 = h_{\text{enc}}(x_0)$ and $\phi_\tau^Z = h_{\text{dyn}} \circ \cdots \circ h_{\text{dyn}}$ (τ times).
4. Given a new state $x \in X$, determine whether it is in the desired RoA using MG.

1. Data Collection The method collects time series data in the form of long trajectories from various initial states in X. The data is partitioned into ordered pairs of the form $(x, \text{Im}(x))$, where $\text{Im}(x) = \phi_\tau(x)$. Denote by $\mathcal{D} = \{(x^i, \text{Im}^i(x))\}_{i=1}^n$ the robot trajectory data collected.

Let $\mathbf{F}$ be the set of all final states of every trajectory in the dataset. If the underlying system has the notion of a "desired" attractor (such as the upright position for a bipedal humanoid), each final state $x_\mathcal{T} \in \mathbf{F}$ is assigned a label $y(x_\mathcal{T})$ of $+1$ if it successfully achieves the desired task or 0 otherwise. This helps determine which attractors discovered by MORALS in the learned latent manifold are desirable.

2. Network Architecture and Training Given a pair $(x, \text{Im}(x))$, the encoder network h_{enc} maps it onto the latent space to obtain $(z, \text{Im}_z(x))$. The decoder network h_{dec} acts on z to obtain $\hat{x}$, which is the *reconstruction* of input state x. The latent dynamics network acts on z to produce the *image* $\text{Im}_z(z)$ in the latent state. The decoder obtains $\widehat{\text{Im}}(x)$, the reconstruction of the input $\text{Im}(x)$ by acting on $\text{Im}_z(z)$.

Training attempts to minimize the following losses.

$$L_1 = \mathbb{E}_{x\sim\mathcal{D}} ||x - h_{\text{dec}}(h_{\text{enc}}(x))||_2^2 \quad (2)$$

$$L_2 = \mathbb{E}_{\text{Im}(x)\sim\mathcal{D}} ||\text{Im}(x) - h_{\text{dec}}(h_{\text{enc}}(\text{Im}(x)))||_2^2 \quad (3)$$

$$L_3 = \mathbb{E}_{(x,\text{Im}(x))\sim\mathcal{D}} ||h_{\text{dyn}}(h_{\text{enc}}(x)) - h_{\text{enc}}(\text{Im}(x))||_2^2 \quad (4)$$

L_1 and L_2 losses enforce that the reconstructed inputs closely match the original ones by minimizing Euclidean distance. L_3 enforces the local dynamics of the latent space by minimizing the Euclidean distance between the latent state image and the encoded image.

All three neural networks are jointly trained to minimize $L = \lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3$, where $\lambda_i, i \in [1, 2, 3]$ are weights for each minimization objective. To ensure that the obtained latent manifold is bounded, the output layers of h_{enc} and h_{dyn} are activated using the $\tanh$ activation. Thus, the obtained latent manifold always lies inside $[-1, 1]^D$.

Optional loss term using labeled data. Split the final states of demonstrated trajectories into desirable and undesirable sets: $\mathbf{F}_\text{s} = \{x \in \mathbf{F} | y(x) = +1\}$ and $\mathbf{F}_\text{f} = \{x \in \mathbf{F} | y(x) = 0\}$. Then, an L_4 loss can separates the encoded final states in $\mathbf{F}_s$ from their counterparts in $\mathbf{F}_f$:

$$L_4 = \mathbb{E}_{x_s\sim\mathbf{F}_s, x_f\sim\mathbf{F}_f}[\sigma(-c||h_{\text{enc}}(x_s) - h_{\text{enc}}(x_f)||)]. \quad (5)$$

To ensure that the encoded final states do not lie on the latent space boundaries, the sigmoid function $\sigma(a) = \frac{1}{1-e^{-a}}$ is used to upper-bound the distance between the encodings of the desirable and undesirable final states. The constant c is a scaling factor. The training procedure iterates between minimizing the loss functions $\lambda_1 L_1 + \lambda_2 L_2 + \lambda_3 L_3$ and $\lambda_4 L_4$.

3. Morse Graph computation in the latent space The uniform discretization $\mathcal{Z}$ of the latent space $[-1, 1]^D$ into $\prod_{i=1}^D 2^{k_i}$ cubes of dimension D needs to be validated since some cubes in $\mathcal{Z}$ may not correspond to a valid state in X when decoded. To ensure a well-defined discretization of Z, $\mathcal{X}$ is validated as follows: A set of valid random points $P_x = \{x_i\}$ in X is encoded to obtain $P_z = \{z_i = h_{\text{enc}}(x_i)\}$. All boxes in $\mathcal{Z}$ that contain at least one point in P_z or are immediate neighbors to a box containing a point in P_z are validated. The set P_x can also use states in the demonstrated trajectory data $\mathcal{D}$.

The *input representation* of the learned dynamics network ϕ_τ^Z is generated by $V(\mathcal{Z})$, the set of all corner points of cubes in $\mathcal{Z}$. The method computes the set of ordered pairs $\Phi_\tau(\mathcal{Z}) := \{(v, \phi_\tau(v)) \mid v \in V(\mathcal{Z})\}$, by calling $h_{\text{dyn}}(v)$ for time τ from all $v \in V(\mathcal{Z})$. Then, $\Phi_\tau(\mathcal{Z})$ is used to generate the *combinatorial representation of the dynamics* of ϕ_τ^Z by approximating it as a *combinatorial multivalued map* $\mathcal{F}$: $\mathcal{Z} \rightrightarrows \mathcal{Z}$, where vertices are n-cubes $\xi \in \mathcal{Z}$. The map $\mathcal{F}$ contains directed edges $\xi \to \xi', \forall\ \xi' \in \Phi_\tau(\xi)$

and the cubes obtained by $\mathcal{F}(\xi)$ are intended to capture the image of $\phi_\tau^Z(\xi)$. $\mathcal{F}$ is computed as follows: Given $z \in Z$, let $\overline{B(z,\delta)} = \{z' \in Z \mid \|z - z'\| \leq \delta\}$ denote the δ-closed ball at state z. Define the *diameter of* $\xi \in \mathcal{Z}$ by $d(\xi) := \max_{z,z' \in \xi} \|z - z'\|$ and the *diameter of* $\mathcal{Z}$ by $d := \max_{\xi \in \mathcal{Z}} d(\xi)$. Note that for a uniform grid, $d = d(\xi)$, independently of the choice of ξ. Let $V(\xi)$ be the set of corner points of the cube ξ and

$$\mathcal{F}(\xi) := \left\{ \xi' \mid \xi' \cap \overline{B\left(\phi_\tau^Z(v), Ld/2\right)} \neq \emptyset \text{ for } v \in V(\xi) \right\} \quad (6)$$

where L is selected to be an upper bound for the Lipschitz constant L_τ of ϕ_τ^Z. Hence, the above definition of $\mathcal{F}$ satisfies:

$$\mathcal{F}_{min}(\xi) := \{\xi' \in \mathcal{X} \mid \xi' \cap \phi_\tau(\xi) \neq \emptyset\} \subset \mathcal{F}(\xi) \quad (7)$$

an *outer approximation* of ϕ_τ^Z. The versatility in defining an outer approximation provides flexibility in incorporating safety constraints by adjusting the parameter L, which bounds the Lipschitz constant L_τ.

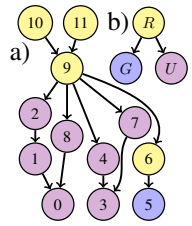

Fig. 4: a) A Morse graph $\mathsf{MG}(\mathcal{F})$ for the humanoid of Fig. 1; b) bistable Morse graph $\mathsf{R_{MG}}$ from Th. 4.1.

In general, no unique Morse graph can be assigned to a dynamical system; instead, the Morse graph provides a rigorous language for describing the structure of the dynamics at different levels of resolution. The proposed method aims to identify whether a given initial condition will lead to success or failure for a provided controller. This can be codified via a Morse graph $\mathsf{R_{MG}} = (\{G, U, R\}, <_r)$ with the partially ordered set (poset) structure $G <_r R$, $U <_r R$, where G denotes success, U denotes failure, and R denotes initial conditions which may lead to success or failure. This Morse graph has two minimal states, G and U.

The number of nodes in the Morse graph $\mathsf{MG}(\mathcal{F})$ computed using Alg. 1 from [3] with inputs $(\mathcal{Z}, \Phi_\tau(\mathcal{Z}), L)$ is typically more than 3 nodes. In particular, it has more than 2 leaf (minimal) nodes, thus providing a richer description of dynamics than required. Nevertheless, as codified in the following theorem, the information from the poset $(\mathsf{MG}(\mathcal{F}), <)$ can be used to produce the desired Morse graph $\mathsf{R_{MG}}$.

Theorem 4.1: Define G as the set of minimal nodes of the Morse graph $\mathsf{MG}(\mathcal{F})$ that correspond to successful final states of the system. Define $R := \{b \in \mathsf{MG} \mid g < b \text{ for some } g \in G\}$ and set $U = \mathsf{MG}(\mathcal{F}) \setminus (G \cup R)$. Then $\mathsf{R_{MG}} = (\{G, U, R\}, <_r)$ with the poset structure $G <_r R$, $U <_r R$ is a Morse graph for the system.

Proof. By [35] it is sufficient to observe (we leave this to the reader) that the map $\rho\colon (\mathsf{MG}(\mathcal{F}), <) \to (\{G, U, R\}, <_r)$ given by identification of elements of $\mathsf{MG}(\mathcal{F})$ with elements of $\mathsf{R_{MG}}$ is an order-preserving map. ■

Fig. 4 depicts the Morse graph $\mathsf{R_{MG}} = (G, U, R, < r)$ from Theorem 4.1, which is obtained from a Morse graph of the humanoid system exemplified by Fig. 1, where $G = \{5\}$, $U = \{0, 1, 2, 3, 4, 7, 8\}$ and $R = \{6, 9, 10, 11\}$. $\mathsf{R_{MG}}$ is also shown in Fig. 1.

4. Obtaining the Regions of Attraction Theorem 1 in [3] guarantees the existence and uniqueness of $\texttt{RoA}(A)$ for attractors A of $\mathsf{MG}(\mathcal{F})$ and Alg. 2 in [3] produces $\texttt{RoA}(A)$. The following theorem ensures that the RoAs found by Algorithm 2 can be used to obtain RoAs associated to the Morse graph $\mathsf{R_{MG}}$ from Theorem 4.1. The proof is a consequence of Alg. 2[3] and Theorem 4.1 here.

Theorem 4.2: Let $\mathcal{F}$ be an outer approximation of the learned dynamics ϕ_τ^Z, $\mathsf{MG}(\mathcal{F})$ be the Morse graph for $\mathcal{F}$, and $\mathsf{R_{MG}} = (\{G, R, U\}, <_r)$ be the Morse graph from Theorem 4.1. Then $\texttt{RoA}(A)$ and $\texttt{RoA}(U)$ computed via $\mathsf{MG}(\mathcal{F})$ are also regions of attraction of G and U in $\mathsf{R_{MG}}$, respectively.

After applying the proposed method to the pedagogical example described in Section III and Fig. 2, the bistability is obtained in the latent space as shown in Fig. 2(right). Notice that the level of discretization required to construct the Morse graph for the original space X is quite extensive, amounting to 5×3^N, approximately 2 million cubes for $N = 12$, which in general is not well suitable for complex, high-dim. systems. When $\mathsf{R_{MG}}$ is computed on the latent space, however, the size of the required discretization is exponentially reduced. For instance, in Fig.2(right) the discretization level is $2K$ cubes. This is a reduction of 3 orders of magnitude compared to the smallest discretization needed on X to find the bistability of Fig. 2(left).

V. Experimental Evaluation

Systems: Fig. 5 shows 3 of the considered systems in the experiments in addition to the humanoid of Fig. 1. In particular, the experiments consider: (1) A model of a **Pendulum** (Pend) observed via the coordinates and velocity of its mass $[x, y, \dot{x}, \dot{y}]$. (2) A **Cartpole** (CaPo) simulated using MuJoCo [36] with state space $[x, \dot{x}, \cos\theta, \sin\theta, \dot{\theta}]$. (3) The **Humanoid** (GetUp) benchmark borrowed from the literature [37] corresponds to a bipedal humanoid robot attempting a stable standup gait. (4) **TriFinger Robot Hand** (TriFi) is a real-world dataset of 3 fingers pushing a cube towards a desired location. [38].

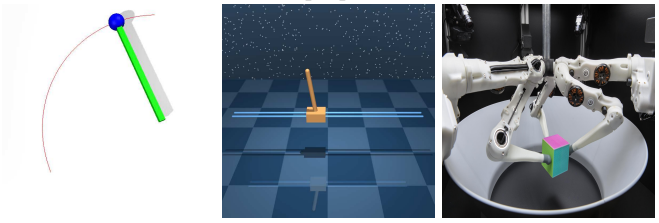

Fig. 5: (L-R) Analytical Pendulum, Cartpole simulated using MuJoCo, real robot dataset collected using a TriFinger [38].

Controllers: Both analytical and learned controllers are considered. LQR linearizes the system to compute a gain k used in the control law $u(x_t) = -kx_t$. It is applied on the Pend and CaPo systems. Soft Actor-Critic (SAC) [39] and Proximal Policy Optimization (PPO) [40] are deep reinforcement learning algorithms trained to maximize the expected return $\mathbb{E}_{x_0}[\sum_{t=0}^{t_{\max}} \gamma^t \mathcal{R}(x_t, u_t)]$, where $\mathcal{R} : X \times U \mapsto \mathbb{R}$ is a reward function that encodes the goal of the desired task and γ is a discount factor. They are applied on the GetUp and TriFi systems, respectively.

Setup: For the simulated systems, the datasets $\mathcal{D}$ are obtained by rolling out trajectories using the controller. For TriFi, the dataset is fixed. Given labeled training and test

datasets ($\mathcal{D}_{tr}, \mathcal{D}_{te}$ - random split 4:1 of trajectories), the autoencoder is trained on $\mathcal{D}_{tr}$ with multiple random seeds for the same hyperparameters and dim(Z) = 2. $h_{\text{enc}}, h_{\text{dec}}, h_{\text{dyn}}$ are fully-connected multi-layered perceptions (MLPs) with either 2 or 3 hidden layers depending on the benchmark. All trials where MORALS discovered less than 2 attractors were restarted. From the set of desirable final conditions in the training set $\mathbf{F}_f^{tr}$ (as in Section IV), all RoAs containing all the points $\{h_{\text{enc}}(x) | x \in \mathbf{F}_f^{tr}\}$ are identified as the desired RoAs. Define $\mathbf{I}_s^{te}$ as the set of initial conditions in $\mathcal{D}_{te}$ that succeed in the task and $\hat{\mathbf{I}}_s^{te}$ as the set of initial conditions in $\mathcal{D}_{te}$ identified by MORALS to be inside the desired RoA. The sets of unsuccessful initial conditions $\mathbf{I}_f^{te}, \hat{\mathbf{I}}_f^{te}$ are similarly defined. Table II reports the precision $\texttt{P} = \frac{|\mathbf{I}_s^{te} \cap \hat{\mathbf{I}}_s^{te}|}{|\hat{\mathbf{I}}_s^{te}|}$, recall $\texttt{R} = \frac{|\mathbf{I}_s^{te} \cap \hat{\mathbf{I}}_s^{te}|}{|\mathbf{I}_s^{te}|}$, and F-score $\texttt{F} = \frac{2\texttt{PR}}{\texttt{P+R}}$. It also reports the number of trajectories $|\pi_{tr}|$ used for training the autoencoder and the number $|\mathcal{D}_{tr}|$ of $(x, \text{Im}(x))$ pairs obtained by applying a sliding window to the trajectory. All metrics are reported on the testing set $\mathcal{D}_{te}$ after selecting the best-performing hyperparameters on $\mathcal{D}_{tr}$.

Benchmark	**dim**(X)	$\lvert\pi_{tr}\rvert$	$\lvert\mathcal{D}_{tr}\rvert$	P	R	F
Pend (LQR)	4	1024	20,480	94%	85%	89%
CaPo (LQR)	5	1440	143,281	86%	77%	81%
GetUp (SAC)	67	1000	326,384	91%	91%	91%
TriFi (PPO)	96	3072	460,806	90%	97%	93%

TABLE II: MORALS performance on benchmarks for dim(Z) = 2.

For Pend (LQR) and CaPo (LQR), $15 - 35\%$ of initial conditions in $\mathbf{I}_s^{tr}$ succeed in the task. The learned controllers for GetUp (SAC) and TriFi (PPO) are more successful ($80 - 90\%$ success rate). Across the benchmarks, MORALS returns rather accurate estimates of the desired RoA.

Data Efficiency: Table III varies the number of trajectories used to train the autoencoding network for the Pend (LQR) benchmark (as a ratio of the available data). Increasing the data used by MORALS during the training phase results in an overall improvement in the desired RoA estimate.

Size	P	R	F
10%	71%	58%	64%
50%	95%	79%	86%
100%	94%	85%	89%

TABLE III: Data-efficiency for RoA estimate of Pend (LQR).

Multiplier	P	R	F
1×	91%	91%	91%
2×	92%	56%	70%
4×	92%	48%	63%

TABLE IV: Ablation on parameter L for GetUp (SAC).

Selecting a suitable L-value: Table IV varies the parameter L that bounds the Lipschitz constant for GetUp (SAC). Increasing L makes the approach more conservative in accepting positives. This improves precision by reducing the occurrence of False Positives near the RoA's boundary. It comes at the cost of reduced recall, however.

	Unlabeled			**Labeled**		
Benchmark	P	R	F	P	R	F
Pend (LQR)	94%	85%	89%	92%	59%	72%
CaPo (LQR)	85%	76%	80%	86%	77%	81%
GetUp (SAC)	91%	91%	91%	82%	76%	79%
TriFi (PPO)	90%	97%	93%	90%	94%	92%

TABLE V: Impact of loss term L_4, which requires labeling.

Effect of supervised loss objective: Table V reports the accuracy of the RoA estimate when the loss function using labeled data as defined in Eqn. 5 is also used. The effect is minimal indicating that MORALS does not need any labels for which demonstration trajectories succeeded or not.

Dimensionality of Latent Space: Tables VI, VII study the impact of $dim(Z)$ for Pend (LQR) and GetUp (SAC). MORALS recovers a significant portion of the RoA for Pend (LQR) even for dim(Z) = 1 but the estimate improves for dim(Z) = 2 as the true dynamics lie in a 2-dim. space. For GetUp (SAC), the R and F scores improve for dim(Z)=3. But this is due to the entire valid Z returned as the RoA leading to more false positives and lower precision.

dim(Z)	P	R	F
1	82%	86%	84%
2	94%	85%	89%

TABLE VI: Impact of dim(Z) for Pend (LQR) RoA.

dim(Z)	P	R	F
2	91%	91%	91%
3	89%	100%	94%

TABLE VII: Impact of dim(Z) for GetUp (SAC) RoA.

Weaker controllers: Table VIII reports accuracy for more complex versions of GetUp (SAC) and TriFi (PPO), where the controller works in $50 - 65\%$ of X. For GetUp, the humanoid's max velocity is limited. So some trajectories are erroneously misclassified as failures as they do not stabilize within the specified horizon, impacting the accuracy of MORALS. The considered version of TriFi requires lifting and transporting the cube to a target instead of the earlier task of pushing. The cube's pose is tracked by noisy cameras, and some critical features may be missing from the dataset resulting in a less accurate h_{dyn}. On these benchmarks, MORALS achieves high recall at the cost of misclassifying multiple regions of the space as false positives.

Benchmark	**dim**(X)	$\lvert\pi_{tr}\rvert$	$\lvert\mathcal{D}_{tr}\rvert$	P	R	F
GetUp (SAC)	67	1000	326,384	78%	96%	86%
TriFi (PPO)	54	1915	574,831	65%	100%	78%

TABLE VIII: Quantitative evaluation for more complex versions of GetUp (SAC) and TriFi (PPO) (dim(Z) = 2).

VI. Discussion

MORALS assumes the dataset $\mathcal{D}_{\text{tr}}$ covers enough initial conditions close to the boundary of success/failure RoAs to approximate the true dynamics. Otherwise, the approach cannot discover the separatrix of the bistable dynamics and discovers only a single attractor. Furthermore, MORALS does not use the decoder h_{dec} to obtain the RoA. Future work will explore using h_{dec} in the high-dim. space with low-accuracy predictions to mitigate False Positives. Finally, MORALS does not provide a conservative estimate of the ground truth RoA due to the stochasticity in training the autoencoder. It is interesting to explore further how to maximize precision.

The RoAs discovered by MORALS for different controllers of the same robotics system can be used to compose a hybrid solution that succeeds in the task from a wider swath of the state space. Given a fixed dataset of robotics transitions, MORALS can discover the regions of the robot's dynamics from where task success is feasible, which has applications in both safe controller learning [41] and safe motion planning [42]. Future directions include learning safety regions for constrained systems [25] and deployment in real robotic systems, where challenges related to system dynamics approximation and data efficiency must be addressed.

REFERENCES

[1] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-Jacobi reachability: A brief overview and recent advances," in *CDC*, 2017.

[2] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-trees: Feedback motion planning via sums-of-squares verification," *IJRR*, vol. 29, no. 8, 2010.

[3] E. R. Vieira, E. Granados, A. Sivaramakrishnan, M. Gameiro, K. Mischaikow, and K. E. Bekris, "Morse Graphs: Topological Tools for Analyzing the Global Dynamics of Robot Controllers," in *The 15th International Workshop on the Algorithmic Foundations of Robotics (WAFR)*, 2022.

[4] E. R. Vieira, A. Sivaramakrishnan, Y. Song, E. Granados, M. Gameiro, K. Mischaikow, Y. Hung, and K. E. Bekris, "Data-Efficient Characterization of the Global Dynamics of Robot Controllers with Confidence Guarantees," in *ICRA*, 2023.

[5] S. Kajita, F. Kanehiro, K. Kaneko, K. Yokoi, and H. Hirukawa, "The 3d linear inverted pendulum mode: A simple modeling for a biped walking pattern generation," in *Proceedings 2001 IEEE/RSJ International Conference on Intelligent Robots and Systems. Expanding the Societal Role of Robotics in the the Next Millennium (Cat. No. 01CH37180)*, vol. 1. IEEE, 2001, pp. 239–246.

[6] H. Geyer, A. Seyfarth, and R. Blickhan, "Compliant leg behaviour explains basic dynamics of walking and running," *Proceedings of the Royal Society B: Biological Sciences*, vol. 273, no. 1603, pp. 2861–2867, 2006.

[7] A. A. Ahmadi, A. Majumdar, and R. Tedrake, "Complexity of ten decision problems in continuous time dynamical systems," in *2013 American Control Conference*. IEEE, 2013, pp. 6376–6381.

[8] P. Giesl and S. Hafstein, "Review on computational methods for Lyapunov functions," *Discrete & Continuous Dynamical Systems-B*, vol. 20, no. 8, p. 2291, 2015.

[9] A. V. Pesterev, "Attraction domain estimate for single-input affine systems with constrained control," *Automation and Remote Control*, vol. 78, no. 4, pp. 581–594, 2017.

[10] ——, "Attraction domain for affine systems with constrained vector control closed by linearized feedback," *Autom. & Remote Control*, vol. 80, no. 5, 2019.

[11] P. A. Parrilo, *Structured semidefinite programs and semialgebraic geometry methods in robustness and optimization*. California Institute of Technology, 2000.

[12] A. Majumdar and R. Tedrake, "Funnel libraries for real-time robust feedback motion planning," *The International Journal of Robotics Research*, vol. 36, no. 8, pp. 947–982, 2017.

[13] A. Vannelli and M. Vidyasagar, "Maximal Lyapunov functions and domains of attraction for autonomous nonlinear systems," *Automatica*, vol. 21, no. 1, pp. 69–80, 1985.

[14] R. Bobiti and M. Lazar, "A sampling approach to constructing lyapunov functions for nonlinear continuous-time systems," in *CDC*, 2016.

[15] S. Chen, M. Fazlyab, M. Morari, G. J. Pappas, and V. M. Preciado, "Learning lyapunov functions for hybrid systems," in *HSCC*, 2021, pp. 1–11.

[16] ——, "Learning region of attraction for nonlinear systems," *arXiv preprint arXiv:2110.00731*, 2021.

[17] G. Mamakoukas, I. Abraham, and T. D. Murphey, "Learning stable models for prediction and control," *IEEE Trans Robot*, 2020.

[18] S. M. Richards, F. Berkenkamp, and A. Krause, "Lyapunov Neural Network: Adaptive stability certification for safe learning of dynamical systems," in *CoRL*, 2018.

[19] A. Lederer and S. Hirche, "Local Asymptotic Stability Analysis and Region of Attraction Estimation with Gaussian Processes," in *CDC*, 2019.

[20] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control barrier functions: Theory and applications," in *2019 18th European Control Conference (ECC)*. IEEE, 2019, pp. 3420–3431.

[21] J. J. Choi, A. Agrawal, K. Sreenath, C. J. Tomlin, and S. Bansal, "Computation of RoAs for Hybrid Limit Cycles Using Reachability," *arXiv:2201.08538*, 2022.

[22] J. H. Gillulay and C. J. Tomlin, "Guaranteed safe online learning of a bounded system," in *IROS*, 2011.

[23] A. K. Akametalu, J. F. Fisac, J. H. Gillula, S. Kaynama, M. N. Zeilinger, and C. J. Tomlin, "Reachability-based safe learning with gaussian processes," in *CDC*, 2014.

[24] L. Wang, E. A. Theodorou, and M. Egerstedt, "Safe learning of quadrotor dynamics using barrier certificates," in *ICRA*, 2018.

[25] C. Dawson, S. Gao, and C. Fan, "Safe control with learned certificates: A survey of neural lyapunov, barrier, and contraction methods for robotics and control," *IEEE Transactions on Robotics*, 2023.

[26] A. Srinivas, A. Jabri, P. Abbeel, S. Levine, and C. Finn, "Universal planning networks: Learning generalizable representations for visuomotor control," in *International Conference on Machine Learning*. PMLR, 2018, pp. 4732–4741.

[27] E. Banijamali, R. Shu, H. Bui, A. Ghodsi *et al.*, "Robust locally-linear controllable embedding," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2018, pp. 1751–1759.

[28] M. Watter, J. Springenberg, J. Boedecker, and M. Riedmiller, "Embed to control: A locally linear latent dynamics model for control from raw images," *Advances in neural information processing systems*, vol. 28, 2015.

[29] B. Ichter and M. Pavone, "Robot motion planning in learned latent spaces," *IEEE Robotics and Automation Letters*, vol. 4, no. 3, pp. 2407–2414, 2019.

[30] N. H. Kim, Z. Xie, and M. van de Panne, "Learning to correspond dynamical systems," in *Learning for Dynamics and Control*, ser. Proceedings of Machine Learning Research, A. M. Bayen, A. Jadbabaie, G. Pappas, P. A. Parrilo, B. Recht, C. Tomlin, and M. Zeilinger, Eds., vol. 120. The Cloud: PMLR, 10–11 Jun 2020, pp. 105–117. [Online]. Available: http://proceedings.mlr.press/v120/kim20a.html

[31] W. D. Kalies, K. Mischaikow, and R. Vandervorst, "Lattice structures for attractors I," *J. Comput. Dyn.*, vol. 1, no. 2, pp. 307–338, 2014.

[32] ——, "Lattice structures for attractors II," *Foundations of Computational Mathematics*, vol. 1, no. 2, pp. 1–41, 2015.

[33] ——, "Lattice Structures for Attractors (III)," *Journal of Dynamics and Differential Equations*, pp. 1572–9222, 2021.

[34] S. Prajna, A. Papachristodoulou, and P. A. Parrilo, "Introducing SOSTOOLS: A general purpose sum of squares programming solver," in *CDC*, 2002.

[35] W. D. Kalies, D. Kasti, and R. Vandervorst, "An algorithmic approach to lattices and order in dynamics," *SIAM Journal on Applied Dynamical Systems*, vol. 17, no. 2, pp. 1617–1649, 2018. [Online]. Available: https://doi.org/10.1137/17M1139606

[36] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, T. Lillicrap, N. Heess, and Y. Tassa, "dm_control: Software and tasks for continuous control," *Software Impacts*, vol. 6, p. 100022, 2020.

[37] T. Tao, M. Wilson, R. Gou, and M. van de Panne, "Learning to get up," in *ACM SIGGRAPH 2022 Conference Proceedings*, ser. SIGGRAPH '22. New York, NY, USA: Association for Computing Machinery, 2022. [Online]. Available: https://doi.org/10.1145/3528233.3530697

[38] N. Gürtler, S. Blaes, P. Kolev, F. Widmaier, M. Wuthrich, S. Bauer, B. Schölkopf, and G. Martius, "Benchmarking offline reinforcement learning on real-robot hardware," in *The Eleventh International Conference on Learning Representations*, 2023. [Online]. Available: https://openreview.net/forum?id=3k5CUGDLNdd

[39] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," in *ICML*, 2018.

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

[41] S. Gu, L. Yang, Y. Du, G. Chen, F. Walter, J. Wang, Y. Yang, and A. Knoll, "A review of safe reinforcement learning: Methods, theory and applications," 2023.

[42] C. Knuth, G. Chou, J. Reese, and J. Moore, "Statistical Safety and Robustness Guarantees for Feedback Motion Planning of Unknown Underactuated Stochastic Systems," *ICRA*, 2023.