V-MORALS: Visual Morse Graph-Aided Discovery of Regions of Attraction in a Learned Space

Anonymous Authors

***Abstract*— Reachability analysis has become increasingly important in robotics to distinguish safe from unsafe states. Unfortunately, existing reachability and safety analysis methods often fall short, as they require known system dynamics or large datasets to estimate accurate system models, are computationally expensive, and assume full state information. A recent tool, called MORALS, aims to address this by using topological tools to estimate Regions of Attraction (ROA) for a controller in a low-dimensional latent space. However, MORALS still relies on full state knowledge and has not been studied when only sensor measurements are available. This paper presents Visual Morse Graph-Aided Discovery of Regions of Attraction in a Learned Space (V-MORALS). V-MORALS takes in a dataset of image-based trajectories of a system under a given controller, and learns a latent space for reachability analysis. Using this learned latent space, our method is able to generate well-defined Morse Graphs, from which we can compute ROAs for various systems and controllers. V-MORALS provides capabilities similar to the original MORALS architecture without relying on state knowledge, and using only high-level sensor data. Our anonymous project website is at: `https://v-morals.onrender.com`.**

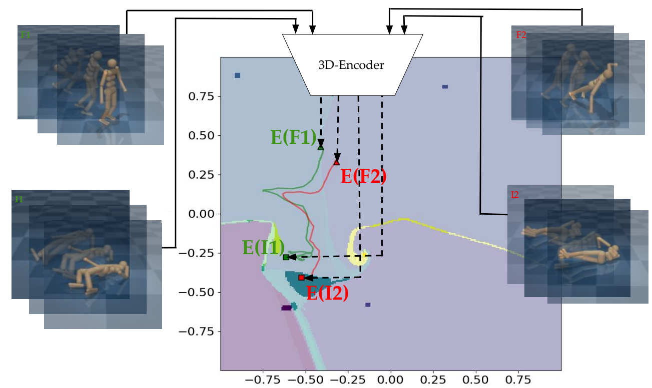

Fig. 1: Region of Attraction (ROA) map of the GetUp controller [10] on a Humanoid. Given a trajectory of images, we learn a latent space to generate a Morse Graph and ROA map. Here, $E(\cdot)$ refers to the encoded latent vector of an image stack. Each colored region of the ROA map corresponds to a different attractor, allowing the system to predict the long-term outcome of a trajectory, such as success $I1 \rightarrow F1$ or failure $I2 \rightarrow F2$, and providing safety analysis without access to the system's state information.

I. Introduction

In current reachability analysis methods, it is difficult to understand the dynamical behavior of a system and controller when the system is high-dimensional or the controller is complex [1], [2]. To analyze such dynamical behavior, we build upon an architecture from [3] called **Mo**rse Graph-aided discovery of **R**egions of **A**ttraction in a learned **L**atent **S**pace (MORALS), to generate a Morse Graph from which we can compute a Region of Attraction (ROA) map [1], [4] in a learned latent space. Morse Graphs [5], [6] provide a powerful way to understand the long-term behavior of complex systems by building a discrete graph of a system's dynamics, which allows us to analyze safety and predict outcomes with mathematical certainty. The derived ROA map is vital to determining the safety of a system because it explains if a robot's trajectories will converge to an equilibrium point (i.e., a safe or failure state of the robot).

Applying a Morse graph directly to high-dimensional systems can be computationally expensive, which is why the MORALS architecture uses a low-dimensional latent space to build the Morse Graph and ROA map. MORALS defines a learned latent space by encoding state information to a lower dimension and understanding the transitions between latent vectors with a latent dynamics network. By generating the Morse Graph and ROA map within this simplified space, MORALS provides an efficient way to analyze the safety of high-dimensional systems and complex controllers.

However, extending this architecture to operate on visual data introduces significant challenges. State representations provide a complete and concise description of a system's configuration, including explicit dynamic variables like joint velocities. In contrast, a single image lacks this motion data, resulting in partial observability and ambiguity, as multiple future states could plausibly follow a single visual frame. Additionally, images are much higher in dimensionality than state-based information. For example, in CartPole, a standard control task in DeepMind Control Suite [7], the state has 4 dimensions, while a corresponding image could be orders of magnitude larger in dimension. When encoding images, latent vectors are unable to store the same level of information as a state-based approach. This also complicates learning dynamics within a latent space [8], as the transition between two latent vectors is only physically meaningful if the corresponding sequence of reconstructed images [9] represents a valid progression in the original environment.

In this paper, we present V-MORALS as a nontrivial extension of MORALS to deal with partial observability. V-MORALS learns system dynamics from visual data (see Figure 1 for an example). To do so, we preprocess each image to generate a binary mask, which isolates the system from the background to reduce input complexity. To embed temporal information, we encode a stack of sequential frames into a single latent vector. This conditions the representation on an initial trajectory and thus constrains the possible future states. We implement this spatiotemporal encoding using a 3D convolutional autoencoder [11]. The encoder compresses the image stack, capturing its visual content and temporal evolution, while the decoder reconstructs it. Both components are trained jointly with a latent dynamics network. Using these networks, V-MORALS effectively captures the underlying dynamics of a controller applied to a system, in a learned latent space.

The contributions of this paper include:

1) V-MORALS extends MORALS by generating Morse Graphs and ROA maps in a latent space with only partial observability. Our approach is able to address the issues of using high-dimensional images and capturing initial dynamics to define a learned latent space.
2) We provide extensive empirical validation of our approach on three standard control benchmarks (Pendulum, CartPole, and Humanoid). Our experiments demonstrate that the model successfully learns the underlying system dynamics and generates accurate Morse Graphs and Regions of Attraction across various controllers and latent space dimensionalities.

II. Related Work

A. Reachability and Safety Analysis

Classical reachability analysis provides formal guarantees about whether trajectories of a dynamical system remain within safe sets or converge to desired equilibria. Tools such as Hamilton-Jacobi reachability have been widely applied in robotics and control to compute forward and backward reachable sets, but can scale poorly due to the exponential dependence on system dimensionality [1], [2], [12]–[14]. Recent work has extended Hamilton–Jacobi reachability to hybrid dynamical systems for walking robots [15] and to adaptive shielding for safe reinforcement learning in real-world robots [16], but these approaches have only been tested on low-dimensional state spaces. Related efforts also explore learning Regions of Attraction for nonlinear systems directly from data [17] with the help of Lyapunov functions [18], [19], but typically to estimate a single attractor.

Complementary approaches based on Control Barrier Functions (CBFs) provide a more tractable alternative, enabling real-time safety filtering of control policies [20]. Recent advances extend these tools by learning CBFs directly from expert demonstrations [21] and by developing model-free formulations that scale to high-dimensional systems without requiring explicit dynamics models [22]. Nevertheless, these methods still assume access to low-dimensional state representations. In contrast, we explore safety analysis in scenarios where explicit system dynamics and full state information are unavailable, requiring us to rely on images.

B. Latent Space Representations for Planning and Control

Learning compact latent representations has become a common strategy for enabling robots to reason from high-dimensional sensory inputs such as images. Rather than operating directly in pixel space, these methods train deep generative or predictive models to capture structure in a lower-dimensional latent space, where dynamics are easier to model and manipulate. This abstraction has improved planning, control, and reinforcement learning [9], [23]–[25].

In robotics, latent representations have been leveraged to support a variety of planning frameworks. For example, latent spaces have guided motion planning [26] and enabled graph-based search for long-horizon planning [8], [27]–[29]. More recent work has applied latent relational dynamics to multi-object manipulation and rearrangement [30]. While these approaches demonstrate the benefit of latent spaces for improving planning and control efficiency, they generally do not analyze system safety. In contrast, our work leverages latent representations as the foundation for estimating Regions of Attraction for safety analysis.

C. Safety and Reachability in Latent Spaces

A growing body of work explores combining safety analysis with latent representations. For example, [31] propose methods for designing controllers directly in latent spaces with provable stability and safety guarantees, enabling formal reasoning without operating in the full high-dimensional observation space. Recent work has also extended reachability analysis to latent spaces to generalize safety beyond collision avoidance, demonstrating that latent models can capture broader classes of constraints relevant for robotics [32].

The most closely-related work to ours is MORALS [3], which combines latent dynamics with Morse Graph analysis to identify attractors and their corresponding ROAs. MORALS provides a powerful framework for checking safety and long-term outcomes, but it assumes access to state information and has not been studied in settings where only raw sensory data is available. In contrast, our work extends MORALS to operate under partial observability using image-based inputs, and thus performs safety analysis into domains where only high-dimensional observations are accessible.

III. Problem Statement and Preliminaries

A. Dynamics and Observation Function

We consider a discrete-time dynamical system with an n-dimensional state space $\mathcal{S} \in \mathbb{R}^n$, governed by a known policy. The system's state $s_t \in \mathcal{S}$ at time t evolves according to the unknown dynamics f such that the next state is $s_{t+1} = f(s_t, u_t)$, where u_t is a control input from a given state or image-based controller and with $s_0 \in \mathcal{S}_{initial}$ being the initial state. While the true state s_t is inaccessible, we can observe the system through an observation function, $\phi : \mathcal{S} \to \mathbb{R}^{H \times W \times C}$ in image space, where H, W, and C indicate the height, width, and number of channels, respectively. The image observation at time t is thus given by $I_t = \phi(s_t)$.

B. Reachable Sets

Our objective is to compute reachable sets having only access to images, and determine if the set of initial images $\mathcal{I}_{initial} = \phi(\mathcal{S}_{initial})$ will result in desirable or undesirable behavior. We compute reachable sets to see how $I_0 \in \mathcal{I}_{initial}$ will converge to an attractor and determine whether it is desirable or undesirable. Conceptually, we consider the image space dynamics $I_{t+1} = g(I_t)$ where g encodes $\phi(f(s_t))$. We also define r as the number of recursive rollout steps of g applied to I_t such that $I_{t+r} = g^r(I_t)$. This also means that $I_t = g^0(I_t)$.

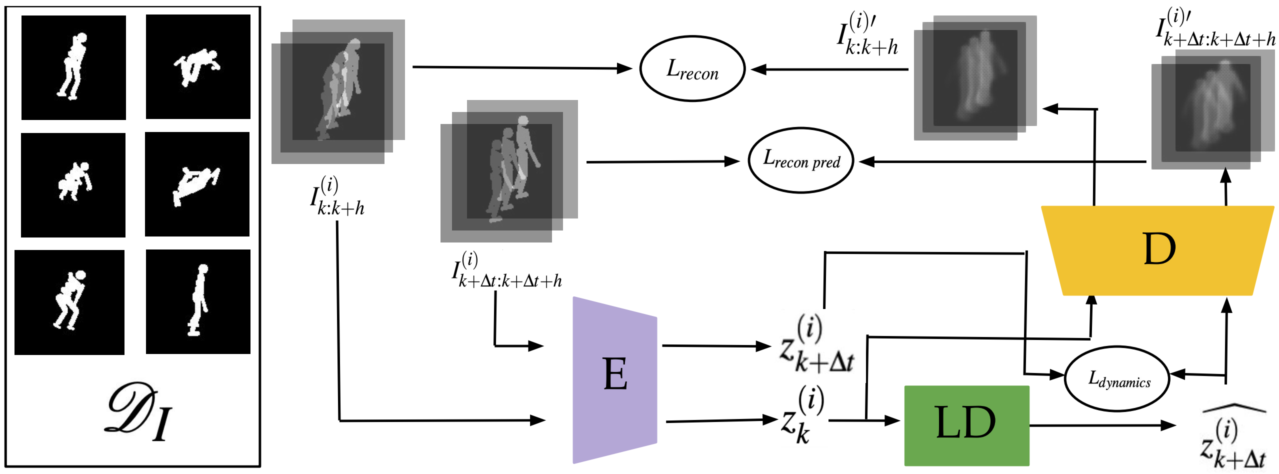

Fig. 2: The V-MORALS architecture and training pipeline. From our dataset $\mathscr{D}_I$ we randomly sample sequence of input images to form each sample, $I_{k:k+h}$. This is mapped to a low-dimensional latent vector z_k by an Encoder (E). A latent dynamics network (LD) is trained to predict the future latent state $\hat{z}_{k+\Delta t}$. A Decoder (D) reconstructs the image sequence $I'_{k:k+h}$ from the latent state z_k. See Section IV for more details.

Given the set of initial observations $\mathscr{I}_{\text{initial}}$, the reachable set, denoted $\mathscr{R}(\mathscr{I}_{\text{initial}})$, is the set of all observations that can be reached from any initial observation in $\mathscr{I}_{\text{initial}}$ over any number of future time steps. Formally, it is the union of all forward images of the initial set under the dynamics g:

$$\mathscr{R}(\mathscr{I}_{\text{initial}}) = \bigcup_{r=0}^{\infty} \{g^r(I_0) \mid I_0 \in \mathscr{I}_{\text{initial}}\}.$$

By computing this set, we can partition the observation space into regions corresponding to distinct long-term behaviors, and identify which initial conditions lead to desirable or undesirable outcomes. We remark that there is a subtle challenge here due to the use of observations. When we observe states directly, i.e., when the observation map ϕ is the identity function, formally defining desirable behaviors is trivial, i.e., reaching a goal region or avoiding obstacles. However, defining desirable behavior in image space is not immediate, highlighting a distinct challenge of our work.

C. Latent Space and Attractors

To analyze the long-term behavior of high-dimensional systems, the MORALS framework [3] leverages tools from combinatorial topology. MORALS creates a finite, discrete representation of a system's dynamics, which allows for the identification of attractors and their corresponding regions of attraction. To provide context for our contributions and establish the necessary background, we outline two key components from MORALS that are central to our image-based adaptation: the use of a learned latent space to manage dimensionality, and the application of Morse Graphs to analyze system dynamics. We can adapt the MORALS framework to our objective by asserting the observation function, ϕ, is the identity.

Even when state measurements are available, a primary challenge in analyzing robotic systems is in the high dimensionality of their state space. MORALS addresses this by learning a low-dimensional latent space, $\mathscr{Z}$, that captures the essential dynamics of the system. This is achieved using an encoder defined as: $E : \mathscr{S} \to \mathscr{Z}$, that maps the high-dimensional state $s \in \mathscr{S}$ to a low-dimensional latent vector $z \in \mathscr{Z}$, and a decoder $D : \mathscr{Z} \to \mathscr{S}$ that reconstructs the original state from the latent vector. By training the encoder and decoder on system trajectories, the framework obtains a compressed representation where the complex dynamics can be analyzed more efficiently.

Once the dynamics are projected into the latent space, MORALS employs tools from combinatorial topology to analyze the behavior of the dynamical system. The central tool for this is the Morse Graph [5], a directed acyclic graph that provides a finite, combinatorial representation of the system's dynamics. The Morse Graph's nodes correspond to the system's recurrent sets, and its edges describe the flow between them. The leaf nodes of this graph represent the system's attractors, which are stable states or limit cycles to which trajectories converge. By identifying attractors, one can then compute their Region of Attraction (ROA), which is the set of all initial states that are guaranteed to converge to a specific attractor. This technique allows MORALS to predict the final outcome (e.g., success or failure) of a trajectory from its initial state, a capability that our work aims to extend to systems observed only through images.

D. Problem Formulation

For a given task, we assume that we are given a dataset of N trajectory rollouts $\mathscr{D} = \{\tau_i\}_{i=1}^N$ where each τ_i consists of a sequence of states: $\tau_i = \{s_0^{(i)}, \ldots, s_k^{(i)}\}$. We assume that these trajectories are obtained from the system under a given controller. We do not need to know the system or the controller, as long as we can collect trajectories. Each image $I_t^{(i)}$ is generated from its corresponding state $s_t^{(i)}$ via the observation function: $\phi(s_t^{(i)}) = I_t^{(i)}$. We also define:

$$\mathscr{I}_i = \{I_0^{(i)}, \ldots, I_k^{(i)}\},$$

as the set of images in trajectory τ_i. We denote our image dataset as $\mathscr{D}_I = \{\mathscr{I}_i\}_{i=1}^N$. To ease notation, we suppress the superscript and use τ when the distinction is not needed.

Lastly, we define y_i as a label describing the trajectory outcome (success or failure) of τ_i.

We demonstrate how we can use this image dataset $\mathcal{D}_I$ to analyze the system's dynamical behavior. Our method trains a model to achieve a dual objective: to predict the eventual outcome of a given trajectory (success or failure), and to generate a high-level, combinatorial map of the state transitions within a learned latent space. See Figure 2 for an overview of the pipeline.

IV. Proposed Method

A. Image-Based Data Generation

For each image in our dataset $\mathcal{D}_I$, we apply a binary mask to isolate the system from the background. This is a crucial preprocessing step because the dynamics of our chosen tasks are governed entirely by the system's physical configuration (i.e., the positions and angles of its parts). The mask removes dynamically irrelevant information, such as texture and lighting, enabling the model to learn a more robust and accurate representation of the system's state.

A key challenge, not present in MORALS [3], is capturing the system's initial dynamics using only image frames. A single image frame may correspond to multiple possible futures, making prediction ambiguous. To capture temporal dynamics, we encode short image sequences, or stacks, into latent vectors. From each image trajectory $\mathcal{I}_i \in \mathcal{D}_I$, we construct a dataset of ordered pairs for training.

Each pair consists of two consecutive image stacks separated by a time interval. Let the stack size (history length) be denoted by h, and the time interval between stack pairs be Δt. A single training pair is then formed by a stack starting at time k and the subsequent stack starting at time $k+\Delta t$:

$$(I^{(i)}_{k:k+h}, I^{(i)}_{k+\Delta t:k+\Delta t+h}),$$

where an image stack $I_{k:k+h}$ is defined as the sequence of h images:

$$I_{k:k+h} = \{I^{(i)}_k, I^{(i)}_{k+1}, \ldots, I^{(i)}_{k+h-1}\}.$$

Each trajectory $\mathcal{I}_i$ is assigned a binary outcome label, $y_i \in \{0,1\}$, corresponding to its final behavior. We define $y_i = 1$ for the trajectories that result in successful completion of the task (a desired outcome) and $y_i = 0$ for those that result in failure (an undesired outcome). The ordered pair with its label is denoted as: $(I^{(i)}_{k:k+h}, I^{(i)}_{k+\Delta t:k+\Delta t+h}, y_i)$.

B. Model Architecture

We define a d-dimensional latent space $\mathcal{Z} \subseteq [-1,1]^d$ using three different networks: the encoder, decoder, and the latent dynamics network. Our encoder and decoder allow us to have meaningful latent representations while our latent dynamics network learns to understand transitions between states in $\mathcal{Z}$. For more details, we refer the reader to MORALS [3].

Our **encoder**, E, is a 3-dimensional convolutional autoencoder [33], designed to process sequences of images. It maps an image stack $I^{(i)}_{k:k+h}$ to a low-dimensional latent vector $z_k \in \mathcal{Z}$, or more formally: $E(I^{(i)}_{k:k+h}) = z^{(i)}_k$. Using 3D convolutions is crucial because they operate across spatial (height, width) and temporal (the stack dimension) axes. This allows the network to directly learn spatiotemporal features, such as motion and velocity, from the raw pixel data.

To enforce the boundaries of the latent space, the final layer of the encoder utilizes a tanh activation function, ensuring all components of the latent vector $z^{(i)}_k$ are mapped to the range $[-1,1]$. The encoder is trained to distill the position of the system and the essential dynamical information from the image sequence into this compact vector.

Our **decoder**, D, is the inverse of E. It maps a latent vector $z_k \in \mathcal{Z}$ from the low-dimensional latent space back to the high-dimensional observation space, reconstructing the original image stack. Architecturally, the decoder mirrors the encoder and uses 3D transposed convolutions. These layers are used for upscaling the compact latent vector back into a full-resolution image stack. This function is defined as: $D(z^{(i)}_k) = I^{(i)\prime}_{k:k+h}$, where $I^{(i)\prime}_{k:k+h}$ is the reconstructed image stack. Since the initial pixel range is $[0,255]$ where 0 is black and 255 is white, we use a sigmoid activation function in the final layer of the decoder. This allows us to produce a binary output which maps each pixel to values $\{0,1\}$ where 0 is still black and 1 is now white. The decoder attempts to reconstruct the original input ($I^{(i)\prime}_{k:k+h} \approx I^{(i)}_{k:k+h}$), which encourages the latent vector z_k to retain salient information.

Lastly, we define our **latent dynamics network**, LD, as a feedforward neural network. This network operates entirely in $\mathcal{Z}$ and is trained to predict the next latent state based on the current one. This function is defined as:

$$LD(z^{(i)}_k) = \widehat{z^{(i)}_{k+\Delta t}}, \tag{1}$$

where $z^{(i)}_k$ is the current latent state and $\widehat{z^{(i)}_{k+\Delta t}}$ is the predicted latent state at the next time step. To ensure the predicted vector remains within the bounds of the latent space, this network also employs a tanh activation function on its final layer. This allows us to simulate and predict the system's behavior within the latent space.

C. Training Objectives

To jointly train the model, our total loss function is composed of four components, which are calculated for each training pair: $(I^{(i)}_{k:k+h}, I^{(i)}_{k+\Delta t:k+\Delta t+h}, y_i)$.

The first component is the autoencoder reconstruction loss, L_{recon}. This loss ensures the encoder-decoder can faithfully compress and reconstruct an image stack. The loss is defined as:

$$L_{recon} = \text{BCE}(I^{(i)}_{k:k+h}, D(E(I^{(i)}_{k:k+h}))). \tag{2}$$

We use the Binary Cross-Entropy (BCE) loss to measure error on our binary images. To compare our original images to the reconstructed images, we divide the original image's pixel values by 255 to make it within 0 (black) and 1 (white).

Second, we compute the latent dynamics loss $L_{dynamics}$. This loss ensures that the difference between our output of the latent dynamics network and our encoded stack $z^{(i)}_{k+\Delta t}$ is minimized. This loss is formally defined as:

$$L_{dynamics} = \text{MSE}(E(I^{(i)}_{k+\Delta t:k+\Delta t+h}), LD(E(I^{(i)}_{k:k+h}))). \tag{3}$$

We use the Mean-Squared Error (MSE) loss function to minimize the Euclidean distance between the two latent vectors in $\mathscr{Z}$.

Third, we compute the reconstruction loss $L_{recon\ pred}$ where we reduce the loss between the second stack in the pair, $I^{(i)}_{k+\Delta t:k+\Delta t+h}$ and the reconstruction of our latent prediction:

$$L_{recon\ pred} = BCE(I^{(i)}_{k+\Delta t:k+\Delta t+h}, D(LD(E(I^{(i)}_{k:k+h})))). \quad (4)$$

Similar to Equation 2, we use BCE to minimize the loss between the two stacks of binary images.

Lastly, we define a contrastive loss, $L_{contrast}$, that structures the latent space by training the model to group latent vectors with the same y_i. This loss operates on a group of latent vectors from the positive class ($\mathscr{Z}_{pos}$) and negative class ($\mathscr{Z}_{neg}$) and is composed of two main objectives:

1) Inter-Class Loss: This component pushes the two clusters apart. It penalizes any pair of positive and negative latent vectors that are closer to each other than a defined margin, m.
2) Intra-Class Loss: This component makes each cluster tighter. It penalizes the pairwise distances between all vectors within the positive cluster and, separately, within the negative cluster.

This loss can be defined as:

$$\begin{aligned} L_{contrast} = & \sum_{z_p \in \mathscr{Z}_{pos}, z_n \in \mathscr{Z}_{neg}} \max(0, m - \|z_p - z_n\|_2^2) \\ & + \sum_{z_{p_i}, z_{p_j} \in \mathscr{Z}_{pos}} \|z_{p_i} - z_{p_j}\|_2^2 \\ & + \sum_{z_{n_i}, z_{n_j} \in \mathscr{Z}_{neg}} \|z_{n_i} - z_{n_j}\|_2^2. \end{aligned} \quad (5)$$

The first term represents the inter-class loss, while the remaining two terms correspond to the intra-class loss for their respective classes. Using these four components we can define our total loss function as follows:

$$L_{total} = \lambda_1 L_{recon} + \lambda_2 L_{dynamics} + \lambda_3 L_{recon\ pred} + \lambda_4 L_{contrast}, \quad (6)$$

where λ_1 through λ_4 are weights chosen to ensure high-fidelity reconstructions from the latent space, while also learning an accurate model of the system's dynamics in $\mathscr{Z}$.

We trained the models (see Section V-B) for each system for 5000 epochs using the Adam optimizer [34] with a learning rate of 1×10^{-4} on a single NVIDIA V100 GPU. The dataset was partitioned into 80% for training and 20% for validation. Our method is not variational [35] as we found that it was not necessary to achieve promising results. The stochastic nature of a variational autoencoder introduces nontrivial challenges for our analysis. Each input would map to a distribution of latent states, increasing the number of potential transitions between cells. This results in a Morse graph with significantly more nodes and complexity, undermining its utility for a high-level, interpretable analysis.

D. Morse Graph and Region of Attraction (ROA) Generation

Using our learned latent space $\mathscr{Z}$, we can construct a Morse Graph MG, which provides a discrete representation of the system's global behavior, and its corresponding ROA map (see Section III-C). We begin by sampling image stacks from the trajectory dataset, and encoding each in $\mathscr{Z}$ using the trained encoder, E. We then discretize this latent space, which is bounded by $[-1,1]^d$, into a grid of hypercubes, or "cells." To ground our analysis in physically meaningful states, we focus only on the cells that contain these encoded data points and their immediate neighbors. We denote this collection of valid cells as C. See Figure 3 for a visualization.

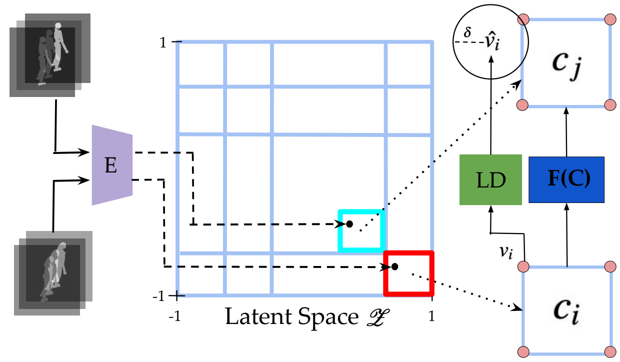

Fig. 3: The process for constructing the directed graph F, which serves as a combinatorial map of the system's dynamics within the learned latent space $\mathscr{Z}$. Image stacks are first mapped into the latent space by the encoder E. This space is then discretized into a grid of cells C. To determine the flow between cells, the corner points of a given cell (c_i) are propagated through the learned dynamics network LD. A safety bubble with radius δ is created around each predicted point to account for prediction uncertainty. A directed edge is drawn from c_i to another cell c_j in the graph $F(C)$ if the union of these safety bubbles intersects with c_j. This process is repeated for all valid cells to build a complete map of the latent dynamics.

To build MG, we determine the transitions between cells in C using our trained latent dynamics model LD. This is achieved by creating a directed graph, F, where each cell is a node. An edge is drawn from a cell c_i to another cell c_j if LD can cause a transition from a state in c_i to a state in c_j. However, since it is computationally infeasible to calculate the future state for every point within a cell, we take the corner points for a given cell $c_i \in C$ and denote the set of these points as V. For a corner point $v_i \in V$, we apply our latent dynamics network to give $LD(v_i) \to \hat{v}_i$ for a certain number of rollout steps. To account for the uncertainty in this prediction and to ensure that we capture all possible future states for any point within a cell c_i, we create a δ-closed ball around $\hat{v}_i$ and each predicted corner point. The radius of this "safety bubble" is determined by the Lipschitz constant L of the dynamics, which provides a mathematical guarantee on the maximum possible divergence of trajectories. A transition from cell c_i to cell c_j is considered possible if the union of these safety bubbles of each predicted corner point intersects with c_j. This process is repeated for all valid cells, resulting in the directed graph F, which serves as a reliable, combinatorial map of the system's dynamics.

With the directed graph F constructed, we now simplify this detailed map of cell-to-cell transitions into the more abstract and interpretable Morse Graph, $MG(F)$. This is a crucial step that distills the complex, cyclical dynamics within F into their essential, long-term behaviors.

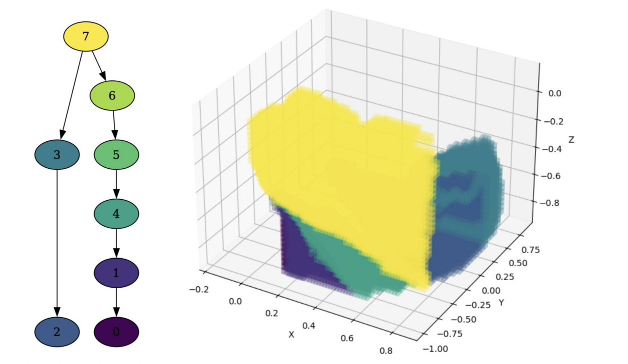

Fig. 4: Morse Graph and ROA map for the Get Up controller applied to a Humanoid, with a latent space dimension of 3. The colors for each Morse Node describe how the set of states changes between each node. The dark blue node represents the attractor of the success region while the dark purple node represents the attractor of the failure. V-MORALS can successfully analyze a complex, high-degree-of-freedom system using only high-dimensional image data, providing an interpretable, low-dimensional map to predict bistable tasks. Additionally, V-MORALS further extends MORALS by being able to visualize ROAs in a 3 dimensional space.

The construction of $MG(F)$ begins by decomposing the graph F into its Strongly Connected Components (SCCs). An SCC is a subgraph where every node, representing a cell in the latent space, is reachable from every other node within that same subgraph. These SCCs represent the recurrent sets of the dynamics, which are regions where the system can cycle indefinitely. Each identified SCC is then collapsed into a single node in a new graph. This new node is called a Morse set. A directed edge is drawn from one Morse set to another if there exists at least one edge in the original graph F from a cell in the first SCC to a cell in the second. The resulting graph is the Morse Graph, $MG(F)$. Since all cycles are contained within each node, the connections between nodes are inherently acyclic. By construction, $MG(F)$ is a directed acyclic graph, which provides a clear, hierarchical representation of state transitions within $\mathscr{X}$.

We then use our Morse Graph to derive the ROA for any given attractor, which corresponds to a leaf node in the graph (see Figure 4). The ROA of an attractor is then defined as the set of cells in C that have a path in F leading to any of the cells within the SCC corresponding to that attractor. By mapping these ROAs, we can partition the state space into distinct basins, providing a formal guarantee on the long-term outcome for any initial state within those regions.

V. Experiments

A. Simulation Tasks

We evaluate our method through three tasks: Pendulum, CartPole, and Humanoid, each with a distinct stabilization goal. The Pendulum's objective is to swing up and balance, the CartPole must balance its pole while moving to the center, and the Humanoid must maintain a standing posture. We select these tasks due to the inherent bistable characteristics of the systems. Predicted trajectory rollouts can therefore be easily classified as either success or failure.

Environment	Latent Dim	Precision	Recall	F-score
Humanoid (SAC)	2	0.9091	0.3846	0.5405
	3	1.0000	0.7253	0.8408
CartPole (LQR)	2	0.2917	0.2979	0.2947
	3	1.0000	0.6809	0.8101
Pendulum (LQR)	2	0.4118	0.8750	0.5600
	3	1.0000	0.4667	0.6364

TABLE I: Performance of V-MORALS in classifying trajectory outcomes across different environments and latent space dimensionalities. For all three metrics (Precision, Recall, and F-score), higher is better.

To get image data, we use the existing MORALS dataset [3] and render state trajectories in MuJoCo [36]. The Humanoid images are rendered using a public repository [10]. For Pendulum and CartPole, we use an LQR controller [37]. For Humanoid, we use a trained Soft Actor-Critic (SAC) [38] policy. Each trajectory in Humanoid, CartPole, and Pendulum has 200, 1000, and 20 frames respectively.

B. Experiment Protocol

We evaluate the effect of latent space dimensionality in three stages. First, we train two distinct models per system, with latent dimensions of 2 and 3. We define a single trajectory instance as a tuple $(I_{0:h}, I_{len(I)-h:len(I)}, y)$, consisting of the initial image stack, the final image stack, and the outcome label ($y=1$ for success, $y=0$ for failure). Using these tuples, we create four sets. Let $B_s = \{E(I_{0:h}) \mid y=1\}$ be the set of initial latent vectors, obtained by encoding the first h frames of every successful trajectory. We also define $B_f = \{E(I_{0:h}) \mid y=0\}$ to be the set of initial latents in each unsuccessful trajectory. Let $L_s = \{E(I_{len(I)-h:len(I)}) \mid y=1\}$ be the set of successful final latent vectors, obtained by encoding the last h frames of successful trajectories only. Similarly we define $L_f = \{E(I_{len(I)-h:len(I)}) \mid y=0\}$.

Second, we generate a Morse Graph and a corresponding Region of Attraction (ROA) map of each system, under both latent dimensions, by simulating the learned dynamics for r rollout steps. The attractor basins in the graph are then labeled: the attractor containing the latent vectors from L_s is designated the "success region," while all other attractors are designated as "failure regions" (see Figure 5).

Finally, we calculate accuracy by classifying the initial states. A prediction for an initial state is correct if its latent vector in B_s maps to the success region or if its latent vector in B_f maps to a failure region. The total accuracy of our model is the fraction of its correctly classified initial states.

C. Results

The performance of our V-MORALS framework in classifying trajectory outcomes is detailed in Table I. The results demonstrate a direct relationship between the dimensionality of $\mathscr{X}$ and the model's predictive accuracy across all three test environments. For each dataset provided by MORALS, 80% was used for training and 20% was used for testing. For the Humanoid system, we built the Morse Graph on latent vectors from the training set. For the Pendulum and CartPole

Benchmark	Model	Precision	Recall	F-Score
Pendulum (LQR)	MORALS	0.9400	0.8500	0.8900
	V-MORALS (Ours)	0.4118	0.8750	0.5600
CartPole (LQR)	MORALS	0.8600	0.7700	0.8100
	V-MORALS (Ours)	0.2917	0.2979	0.2947
Humanoid (SAC)	MORALS	0.9100	0.9100	0.9100
	V-MORALS (Ours)	0.9091	0.3846	0.5405

TABLE II: Comparison of V-MORALS against the original state-based MORALS framework [3] where both use a latent dimension of two. For Precision, Recall, and F-score metrics, higher is better.

Model	Latent Dim	Precision	Recall	F-Score
MORALS (State-based)	2	0.9100	0.9100	0.9100
	3	0.8900	1.0000	0.9400
V-MORALS (Ours)	2	0.9091	0.3846	0.5405
	3	1.0000	0.7253	0.8408

TABLE III: Performance comparison on the Humanoid (GetUp) benchmark [10] between the state-based MORALS and our image-based V-MORALS across latent dimensions.

environments, where the datasets were sparse, we sampled points from $\mathscr{X}$ to build each Morse Graph.

A primary finding is the universal performance improvement when increasing the latent dimension from 2 to 3. In the CartPole environment, the F-score increased substantially from 0.2947 to 0.8101. We also observe a similar gain in Humanoid, from 0.5405 to 0.8408. This trend suggests that a 2-dimensional latent space is insufficient to capture the complexity of the system dynamics required for accurate outcome prediction, whereas a 3-dimensional space provides a significantly richer and more effective representation. Qualitatively, we find that the Morse Graph is simpler (with fewer nodes), and encompasses the bistable nature of the task (see Figure 6), when increasing the latent dimension. The magnitude of this improvement also correlates with trajectory length. V-MORALS significantly improved in predicting outcomes of initial states for CartPole, which has 1000 frames per trajectories, after increasing the dimensionality. In contrast, V-MORALS saw a much smaller gain with the Pendulum environment, which only has 20 frames per trajectory. This indicates that longer trajectories better leverage the increase in latent dimensions.

To contextualize the performance of our image-based method, Table II provides a direct comparison between V-MORALS (using a 2D latent space) and the original MORALS framework which operates on true state information. The performance at a latent dimension of 2 establishes a clear baseline, demonstrating that while the task is inherently more difficult, our approach is viable. This motivates our results in Table III, which show that this performance gap can be closed by increasing the latent space dimensionality to better capture the system's complex dynamics. For better visualizations of ROA maps, we refer the reader to our website: https://v-morals.onrender.com.

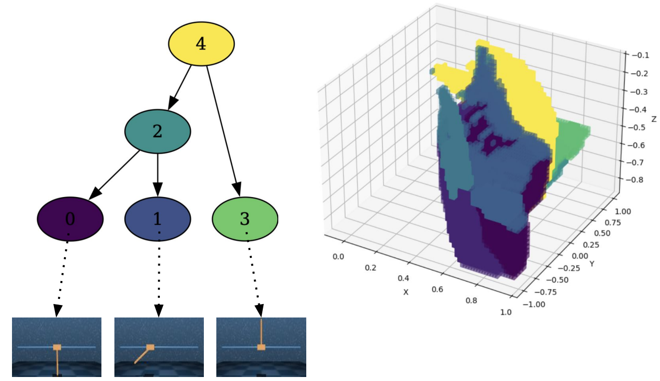

Fig. 5: Analysis of the CartPole system's dynamics in a 3-dimensional learned latent space. (Left) The generated Morse Graph identifies three leaf nodes. The images below show the corresponding physical outcomes: the desired balanced state (green node) and two distinct failure modes (light purple, pole falling left; dark purple, pole straight down). (Right) The ROA map partitions the latent space, where each colored region represents the set of initial states guaranteed to converge to its respective attractor.

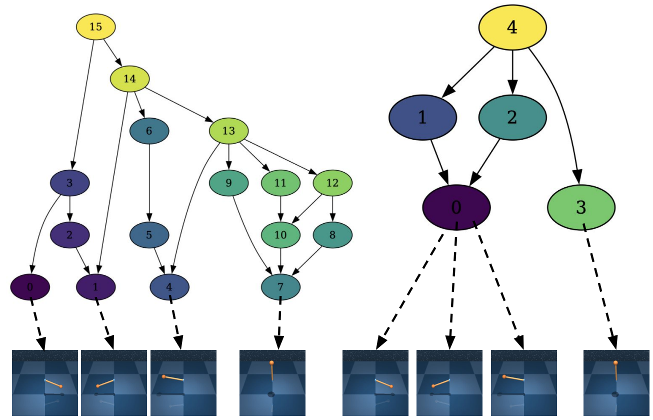

Fig. 6: A comparison of Morse Graphs generated for Pendulum using different latent space dimensions. (Left) A 2-dimensional latent space yields a complex graph with multiple attractors (leaf nodes), failing to capture the true bistable nature of the task. (Right) Increasing the latent dimension to 3 results in a simpler and more accurate topological representation. The dynamics correctly converge to two distinct attractors (dark purple and green), which correspond to the task's success and failure states.

VI. Limitations

While a promising approach, V-MORALS has some limitations that motivate opportunities for future work. First, our method relies on images being a relatively complete representation of what the robot is doing, and may struggle if images contain significant partial observability. Second, our method, as with the original MORALS, assumes that there are fixed regions of attraction, which might not exhaustively characterize all robotics tasks. Finally, we only test on a set of simulated tasks, and in future work we plan to test this using images from real-world robotics tasks, as well as to explore different types of manipulators to understand the cross-embodiment transfer [39] of latent space analysis.

VII. Conclusion

This paper presents Visual Morse Graph-Aided discovery of Regions of Attraction in a learned Latent Space (V-MORALS). V-MORALS provides capabilities similar to the original MORALS architecture without relying on state knowledge, or the controller used for the system. Our method is able to generate well-defined Morse Graphs and ROAs from different controllers for the Pendulum, CartPole, and Humanoid environments. We hope that this work inspires future research in the analysis of reachability and safety of controllers for complex, high-dimensional scenarios.

References

[1] S. Bansal, M. Chen, S. Herbert, and C. J. Tomlin, "Hamilton-Jacobi Reachability: A Brief Overview and Recent Advances," in Conference on Decision and Control (CDC), 2017.

[2] I. M. Mitchell, A. M. Bayen, and C. J. Tomlin, "A time-dependent Hamilton-Jacobi formulation of reachable sets for continuous dynamic games," IEEE Transactions on Automatic Control, 2005.

[3] E. R. Vieira, A. Sivaramakrishnan, S. Tangirala, E. Granados, K. Mischaikow, and K. E. Bekris, "MORALS: Analysis of High-Dimensional Robot Controllers via Topological Tools in a Latent Space," in IEEE International Conference on Robotics and Automation (ICRA), 2024.

[4] R. Tedrake, I. R. Manchester, M. Tobenkin, and J. W. Roberts, "LQR-Trees: Feedback Motion Planning via Sums-of-Squares Verification," in International Journal of Robotics Research (IJRR), 2010.

[5] 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 Workshop on the Algorithmic Foundations of Robotics (WAFR), 2022.

[6] 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 IEEE International Conference on Robotics and Automation (ICRA), 2023.

[7] Y. Tassa, Y. Doron, A. Muldal, T. Erez, Y. Li, D. de Las Casas, D. Budden, A. Abdolmaleki, J. Merel, A. Lefrancq et al., "Deepmind control suite," arXiv preprint arXiv:1801.00690, 2018.

[8] M. Lippi, M. C. Welle, A. Gasparri, and D. Kragic, "Ensemble Latent Space Roadmap for Improved Robustness in Visual Action Planning," in IEEE International Conference on Robotics and Automation (ICRA), 2024.

[9] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson, "Learning Latent Dynamics for Planning from Pixels," in International Conference on Machine Learning (ICML), 2019.

[10] T. Tao, M. Wilson, R. Gou, and M. van de Panne, "Learning to Get Up," in ACM SIGGRAPH Conference, 2022.

[11] D. Tran, L. Bourdev, R. Fergus, L. Torresani, and M. Paluri, "Learning Spatiotemporal Features with 3D Convolutional Networks," in IEEE/CVF International Conference on Computer Vision (ICCV), 2015.

[12] M. Chen, S. L. Herbert, M. S. Vashishtha, S. Bansal, and C. J. Tomlin, "Decomposition of Reachable Sets and Tubes for a Class of Nonlinear Systems," IEEE Transactions on Automatic Control, 2018.

[13] I. M. Mitchell, "A Toolbox of Level Set Methods," Department of Computer Science, University of British Columbia, 2007.

[14] S. Bansal and C. Tomlin, "DeepReach: A Deep Learning Approach to High-Dimensional Reachability," in IEEE International Conference on Robotics and Automation (ICRA), 2021.

[15] J. J. Choi, A. Agrawal, K. Sreenath, C. J. Tomlin, and S. Bansal, "Computation of Regions of Attraction for Hybrid Limit Cycles Using Reachability: An Application to Walking Robots," in IEEE Robotics and Automation Letters (RA-L), 2022.

[16] M. Lu, J. S. Gosain, L. Sang, and M. Chen, "Safe Learning in the Real World via Adaptive Shielding with Hamilton-Jacobi Reachability," in Learning for Dynamics and Control (L4DC), 2025.

[17] S. Chen, M. Fazlyab, M. Morari, G. J. Pappas, and V. M. Preciado, "Learning Region of Attraction for Nonlinear Systems," in Conference on Decision and Control (CDC), 2021.

[18] S. M. Richards, F. Berkenkamp, and A. Krause, "The Lyapunov Neural Network: Adaptive Stability Certification for Safe Learning of Dynamical Systems," in Conference on Robot Learning (CoRL), 2018.

[19] A. Lederer and S. Hirche, "Local Asymptotic Stability Analysis and Region of Attraction Estimation with Gaussian Processes," in Conference on Decision and Control (CDC), 2019.

[20] A. D. Ames, S. Coogan, M. Egerstedt, G. Notomista, K. Sreenath, and P. Tabuada, "Control Barrier Functions: Theory and Applications," in European Control Conference (ECC), 2019.

[21] A. Robey, H. Hu, L. Lindemann, H. Zhang, D. V. Dimarogonas, S. Tu, and N. Matni, "Learning Control Barrier Functions from Expert Demonstrations," in Conference on Decision and Control (CDC), 2020.

[22] D. D. Oh, J. Lidard, H. Hu, H. Sinhmar, E. Lazarski, D. Gopinath, E. S. Sumner, J. A. DeCastro, G. Rosman, N. E. Leonard, and J. F. Fisac, "Safety with Agency: Human-Centered Safety Filter with Application to AI-Assisted Motorsports," in Robotics: Science and Systems (RSS), 2025.

[23] D. Hafner, T. Lillicrap, M. Norouzi, and J. Ba, "Mastering Atari with Discrete World Models," in International Conference on Learning Representations (ICLR), 2021.

[24] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi, "Dream to Control: Learning Behaviors by Latent Imagination," in International Conference on Learning Representations (ICLR), 2020.

[25] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap, "Mastering diverse control tasks through world models," Nature, vol. 619, no. 7969, pp. 360–367, 2023.

[26] B. Ichter and M. Pavone, "Robot Motion Planning in Learned Latent Spaces," in IEEE Robotics and Automation Letters (RA-L), 2019.

[27] M. Lippi, P. Poklukar, M. C. Welle, A. Varava, H. Yin, A. Marino, and D. Kragic, "Latent Space Roadmap for Visual Action Planning of Deformable and Rigid Object Manipulation," in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2020.

[28] M. Lippi, M. C. Welle, P. Poklukar, A. Marino, and D. Kragic, "Augment-Connect-Explore: a Paradigm for Visual Action Planning with Data Scarcity," in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2022.

[29] M. Lippi, M. C. Welle, M. Moletta, A. Marino, A. Gasparri, and D. Kragic, "Visual Action Planning with Multiple Heterogeneous Agents," in IEEE International Conference on Robot and Human Interactive Communication (ROMAN), 2024.

[30] Y. Huang, N. C. Taylor, A. Conkey, W. Liu, and T. Hermans, "Latent Space Planning for Multi-Object Manipulation with Environment-Aware Relational Classifiers," in IEEE Transactions on Robotics (T-RO), 2024.

[31] P. Lutkus, K. Wang, L. Lindemann, and S. Tu, "Latent Representations for Control Design with Provable Stability and Safety Guarantees," arXiv preprint arXiv:2505.23210, 2025.

[32] K. Nakamura, L. Peters, and A. Bajcsy, "Generalizing Safety Beyond Collision-Avoidance via Latent-Space Reachability Analysis," in Robotics: Science and Systems (RSS), 2025.

[33] N. Ravi, J. Reizenstein, D. Novotný, T. Gordon, W. Lo, J. Johnson, and G. Gkioxari, "Accelerating 3d deep learning with PyTorch3D," arXiv preprint arXiv:2007.08501, 2020.

[34] D. P. Kingma and J. Ba, "Adam: A Method for Stochastic Optimization," in International Conference on Learning Representations (ICLR), 2015.

[35] D. P. Kingma and M. Welling, "Auto-Encoding Variational Bayes," in International Conference on Learning Representations (ICLR), 2014.

[36] E. Todorov, T. Erez, and Y. Tassa, "MuJoCo: A Physics Engine for Model-based Control," in IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), 2012.

[37] B. D. O. Anderson and J. B. Moore, Optimal Control: Linear Quadratic Methods. Prentice Hall, 1990.

[38] T. Haarnoja, A. Zhou, P. Abbeel, and S. Levine, "Soft Actor-Critic: Off-Policy Maximum Entropy Deep Reinforcement Learning with a Stochastic Actor," in International Conference on Machine Learning (ICML), 2018.

[39] E. Bauer, E. Nava, and R. K. Katzschmann, "Latent Action Diffusion for Cross-Embodiment Manipulation," arXiv preprint arXiv:2506.14608, 2025.