Learning Reusable Behavioral Motifs for Continuous Animal Behavior Modeling

Jiyi Wang, Jingyang Ke, Bo Dai, Anqi Wu
School of Computational Science and Engineering
Georgia Institute of Technology, Atlanta, GA 30332
{jwang3617@, jingyang.ke@, bodai@cc., awu36@}gatech.edu

Abstract

Natural behavior evolves continuously and often reuses movement structure across contexts, but existing behavioral segmentation methods typically assign a single discrete syllable at each time point or impose restrictive generative assumptions. We introduce Motif-based Continuous Dynamics (MCD), a computational framework that models behavior as time-varying combinations of reusable behavioral motifs. MCD learns motifs as latent behavior-level features from transition dynamics, through spectral representation learning within a reinforcement learning framework. MCD then infers policies that combine these motifs across time or context, through offline imitation learning. This formulation avoids hard behavioral boundaries while capturing continuity and compositionality. Across simulated gridworld and continuous-control tasks, mouse maze navigation and pose data, MCD identifies reusable motifs that explain task and temporal variation. MCD therefore provides a flexible and interpretable framework for computational behavior analysis. Code is available in `https://github.com/G1NO3/MCD`.

1 Introduction

A central goal in animal behavior research is to discover structured representations of behavior from large-scale recordings. Existing approaches often do so by identifying recurring behavioral syllables—such as grooming, running, or sniffing—and partitioning continuous trajectories into discrete units [1, 2, 3, 4, 5]. Despite their success in identifying structured behavioral patterns, these methods typically assign a single behavioral label to each time point and impose a fundamentally discrete view of behavior.

Two important properties of natural behavior are difficult to capture under this discrete formulation. First, **continuity**, real behavior evolves smoothly over time, whereas discrete segmentation introduces artificial boundaries and ambiguity during transitions. Second, **compositionality**, multiple movement tendencies may occur simultaneously, and related behaviors may reuse common components. For example, back and side grooming may share similar forelimb motions while differing in body orientation, and sniffing may co-occur with distinct locomotor patterns like walking or sitting. A single discrete label cannot directly represent such overlap. In addition, many approaches are either non-generative, such as clustering-based methods ([4, 5]), or rely on restrictive assumptions like linear or Markovian dynamics ([1, 2, 3]), which may limit the behaviors they can represent or generate. These limitations motivate modeling behavior as a continuous control process rather than exclusively as a sequence of discrete states.

To address these limitations, we propose a framework for continuous behavior modeling based on reinforcement learning (RL). We view behavior as arising from an underlying policy and infer this policy from fixed behavioral trajectories through offline imitation learning. Rather than treating

40th Conference on Neural Information Processing Systems (NeurIPS 2026).

behavior as transitions between discrete syllables, our framework represents behavioral dynamics through continuously varying policies.

Motivated by the hypothesis that animals generate behavior by recombining a finite set of reusable components over time [6, 7], we introduce *Motif-based Continuous Dynamics discovery (MCD)*. Operationally, MCD defines a behavioral motif as a reusable, behavior-level latent state–action feature learned from observed transition dynamics. We use spectral representation learning within the RL framework [8, 9, 10] to learn the shared motifs $\phi(s, a)$, and offline imitation learning to estimate coefficients u that combine these motifs to parameterize the policy through $Q = \phi(s, a)^\top u$. The behavioral pattern captured by a motif could be low-level movement kinematics or high-level navigation strategies, depending on the state and action representations selected for the scientific question. For example, a free-moving mouse may show both face grooming and side grooming behaviors. We may infer common "grooming" motifs and different motifs involving different body parts. Our intended interpretation here is at the behavioral level, and connections to muscle synergies or neural motor primitives are presented in the Discussion.

A key feature of MCD is that the policy is represented as a continuous combination of reusable motifs whose contributions can vary across tasks or evolve smoothly over time. Multiple motifs can be active simultaneously, allowing complex behaviors to emerge through dynamic composition rather than exclusive discrete switching. For example, side grooming may combine a "grooming" motif with a "turning" motif (Fig. 1). Therefore, this formulation naturally captures continuity and compositionality.



Figure 1: MCD models behavior as continuous compositions of reusable motor motifs. For example, “side grooming” may emerge from the combination of grooming and turning-side motifs.

Taken together, MCD provides a unified computational framework for modeling animal behavior as continuously evolving combinations of reusable behavioral motifs. Across simulated gridworld and continuous-control tasks, mouse maze navigation behavior, free-moving behavior, and macaque drawing behavior, MCD identifies reusable motifs that capture continuity and compositionality. These results support MCD as a flexible and interpretable framework for computational behavior analysis.

2 Related work

Animal behavior analysis. A major direction in animal behavior research is to discover reusable behavioral structure from large-scale recordings. Existing approaches typically do so by identifying recurring behavioral syllables, such as grooming or locomotion, using supervised classification (SIPEC [11], MARS [12]), clustering-based methods (B-SOiD [4], MotionMapper [5], SemiSeg [13]), or hidden Markov model (HMM)-based approaches (VAME [3], MoSeq [1], Keypoint-MoSeq [2], WARHMM [14]). Among them, Keypoint-MoSeq [2] is based on the autoregressive HMM (AR-HMM) method and has become a widely used modern framework in animal behavior analysis. These methods have been successful in uncovering stereotyped behavioral patterns and relating them to neural activity. However, they generally model behavior through discrete latent states, making it difficult to capture continuously evolving and compositional dynamics across contexts and timescales.

A closely related direction applies reinforcement-learning-based imitation learning to animal behavior analysis [15]. Unlike our work, however, these approaches do not explicitly decompose behavior into compact and interpretable motor motifs.

A key methodological distinction from prior behavior modeling approaches lies in how behavioral dynamics are represented. Existing dynamics-based methods often model state transitions $P(s'|s)$ directly through simplified assumptions such as Gaussian or Markovian dynamics [1, 2]. In contrast, MCD models behavior through policy-conditioned dynamics within an RL framework, where transitions emerge from the interaction between environment dynamics $P(s'|s, a)$ and behavior policies $\pi(a|s)$ as $P(s'|s) = \sum_a P(s'|s, a)\,\pi(a|s)$. This formulation provides a more flexible representation of behavior while avoiding strong assumptions on state transitions.

Reinforcement learning. Reinforcement learning has been widely used for skill discovery and hierarchical control in robotics and embodied agents [16, 17, 18]. However, these approaches are primarily designed for control optimization rather than scientific analysis of natural behavior. In

contrast, we use RL as a framework for discovering interpretable and reusable behavioral motifs from offline animal behavior data.

Our work is particularly related to offline skill discovery methods such as OPAL [18] and SPiRL [19], which learn latent skill representations from demonstrations. Unlike these approaches, MCD represents policies through generalized linear combinations of shared motifs, providing more interpretable behavioral representations. Additional RL-related work is provided in Appendix O.

3 Preliminaries

Markov Decision Processes (MDP). To formalize motifs underlying behavior, we model behavioral trajectories within the framework of Markov Decision Processes (MDPs). An MDP is defined as a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, P, \rho, \gamma, H)$, where $\mathcal{S}$ denotes the state space (e.g., pose keypoints capturing the animal's condition and environment), $\mathcal{A}$ the action space that describes feasible movements, $r : \mathcal{S} \times \mathcal{A} \to [0, 1]$ the reward function that encodes the immediate utility toward an internal goal, $P : \mathcal{S} \times \mathcal{A} \to \Delta(\mathcal{S})$ the transition kernel, with $\Delta(\mathcal{S})$ representing distributions over $\mathcal{S}$; $\rho \in \Delta(\mathcal{S})$ the initial state distribution, $\gamma \in (0, 1)$ the discount factor, and H the horizon. A policy $\pi : \mathcal{S} \times [H] \to \Delta(\mathcal{A})$ specifies a distribution over actions given the current state. We assume that observed behavior is generated by a policy optimizing an intrinsic reward that reflects internal motivations. Under this view, behavioral trajectories arise from the interaction between the policy and environment dynamics.

Following standard notations, we define the value function $V(s) := \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) | s_0 = s\right]$ and the action-value function $Q(s, a) = \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) | s_0 = s, a_0 = a\right]$, the expected discounted cumulative rewards under policy π. These satisfy the Bellman equations:

$$Q_h(s, a) = r(s, a) + \gamma\, \mathbb{E}_{s' \sim P(\cdot|s,a)}\big[V_{h+1}(s')\big], \quad V_h(s) = \mathbb{E}_{a \sim \pi(\cdot|s)}\big[Q_h(s, a)\big]. \tag{1}$$

Offline Imitation Learning. We adopt an offline imitation learning (IL) setting to infer a policy π that matches observed behavior. Instead of interacting with the environment, we are given a dataset of transitions collected from an expert (the animal): $\mathcal{D} = \{(s_i, a_i, s'_i) | (s, a) \sim \tau^e, s' \sim P(\cdot|s, a), i = 1, 2, ..., N\}$, where τ^e denotes the empirical state–action distribution. The goal is to recover a policy consistent with this dataset.

4 Methods

Given the MDP formulation, we frame motif discovery from a control-theoretic perspective. Motifs serve as fundamental building blocks that enable the construction of policies and reward functions, thereby providing a compact representation of behavioral dynamics.

Definition 1 (Motif Set)**.** *A transition kernel $P(s'|s, a)$ can be expressed via a spectral decomposition:*

$$P(s'|s, a) = \phi(s, a)^\top \mu(s') q(s'), \tag{2}$$

*where $\phi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^d$, $\mu : \mathcal{S} \to \mathbb{R}^d$, and $q \in \Delta(\mathcal{S})$ is a parametrized probability distribution over the state space. We define ϕ as the **motif** set, and parameterize the reward as $r(s, a) = \phi(s, a)^\top w$.*

Spectral decomposition is a standard tool in RL representation learning [9, 10, 20]. Here, it provides a way to extract latent features directly from transition kernel, without imposing extra explicit assumptions. As a result, the learned motifs reflect intrinsic structure in the behavior data.

Given the motif definition, substituting Eq. 2 and the linear reward model into Eq. 1, we get:

$$Q(s, a) = r(s, a) + \gamma \int V(s') P(s'|s, a) \mathrm{d}s' = \phi(s, a)^\top \left[w + \gamma \int V(s') \mu(s') q(s') \mathrm{d}s' \right] = \phi(s, a)^\top u, \tag{3}$$

where $u = w + \gamma \int V(s')\, \mu(s')\, q(s')\, \mathrm{d}s'$. Thus, $Q(s, a)$ can be expressed as a linear combination of motif features $\phi(s, a)$, linking motifs to the policy. Following the maximum entropy RL framework [21], the optimal policy takes the form:

$$\pi(a|s) = \arg\max_{\pi}\ \mathbb{E}_\pi[Q(s, a)] + H(\pi) = \arg\max_{\pi} \frac{\exp(\phi(s, a)^\top u)}{\sum_{a' \in \mathcal{A}} \exp(\phi(s, a')^\top u)}, \tag{4}$$

where $H(\pi) := -\sum_{a \in \mathcal{A}} \pi(a|s)\ \log\big(\pi(a|s)\big)$ is the entropy.

Proposition 1. *The policy in Eq. 4 arises as the maximum-entropy solution induced by the spectral representation of the transition kernel. The motifs ϕ form a basis that can represent any max-entropy policy through an appropriate choice of u.*

We interpret $\phi(s, a)$ as motifs because it provides a shared basis for transitions, policies, and rewards. Policies describe how behavior evolves, while rewards capture why it evolves. Since ϕ is derived solely from transition dynamics, it encodes intrinsic, reusable motor structure independent of specific tasks. The coefficients u capture task-dependent modulation, yielding an interpretable decomposition of behavior as compositions of shared motifs. Unlike prior dynamics-based approaches that directly model $P(s'|s)$ under restrictive assumptions, our formulation represents behavior through a shared motif basis, avoiding explicit parametric dynamics models.

Together, Def. 1 and Prop. 1 show that our framework captures behavioral dynamics without relying on restrictive model assumptions, in contrast to clustering-based methods (which lack dynamics) and traditional dynamics models (which impose strong assumptions). Next, we describe how to learn $\phi(s, a)$ and $\mu(s')$ (motif discovery) and u (policy learning) from data. The learning procedure differs between discrete and continuous settings.

4.1 Discrete Version

Motif discovery. For discrete state-action spaces, motif representations can be learned via (1) classical spectral methods such as singular value decomposition (SVD) [22, 23, 24], or (2) spectral decomposition–based representation learning [9, 25]. We adopt the latter and optimize

$$\phi(s, a), \mu(s') = \arg\min_{\phi,\mu} \left\| P(s'|s, a) - \phi(s, a)^\top \mu(s') q(s') \right\|^2 . \tag{5}$$

We set the auxiliary distribution to $q(s') = \rho(s') = \int \tau^e(s, a)\, da$, the marginalized state distribution. The learned motif set $\phi(s, a)$ is then used for policy learning.

Motif-based policy learning. We estimate the policy $\pi(a|s)$ (Eq. 4) via maximum likelihood estimation (MLE), i.e., by optimizing the following objective to solve for u:

$$\max_u \ \mathbb{E}_{(s,a)\sim\tau^e} \left[\log \pi(a|s)\right] = \max_u \ \mathbb{E}_{(s,a)\sim\tau^e} \left[\log \frac{\exp(\phi(s, a)^\top u)}{\sum_{a'\in\mathcal{A}} \exp\left(\phi(s, a')^\top u\right)}\right] . \tag{6}$$

4.2 Continuous Version

The continuous setting introduces two key challenges. First, the decomposition in Eq. 2 is too restrictive to capture realistic behavioral dynamics. For example, a common model assumes additive structure, $s' = h(s, a) + \epsilon$, where h is a dynamics function and ϵ is Gaussian noise. This may not be well captured by the multiplicative form $\phi(s, a)^\top \mu(s') q(s')$, suggesting the need for a more flexible parameterization.

While learning from discrete data is relatively straightforward, the continuous setting introduces two key challenges. First, the decomposition in Eq. 2 is too restrictive to capture realistic behavioral dynamics. For example, consider a common and biologically plausible additive behavioral dynamics, $s' = h(s, a) + \epsilon$, where h is a dynamics function and ϵ is Gaussian noise. This dynamics contrast with the multiplicative form $\phi(s, a)^\top \mu(s') q(s')$, suggesting the need for a more flexible parameterization. Second, policy learning becomes intractable: the partition function in Eq. 6 requires integration over a continuous action space with infinite actions that are impossible to enumerate. To address these challenges, we adopt an alternative formulation.

Motif discovery. We model the transition kernel $P(s'|s, a)$ as an energy-based model (EBM) [10]:

$$P(s'|s, a) = q(s') \exp\left(\psi(s, a)^\top \nu(s') - \log Z(s, a)\right), \quad Z(s, a) = \int q(s') \exp(\psi(s, a)^\top \nu(s'))ds', \tag{7}$$

where $\psi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}^g$ and $\nu : \mathcal{S} \to \mathbb{R}^g$ are neural-network feature maps and $Z(s, a)$ is the partition function. Compared to Eq. 2, this EBM formulation yields more expressive motif representation and also produces more stable gradients during training.

Proposition 2 (Connection to Motif Definition)**.** *The EBM formulation induces a representation equivalent to Eq. 2, where $\phi(s, a)$ and $\mu(s')$ are functions of ψ, ν, and $Z(s, a)$. (Appendix G)*

We learn ψ and ν using noise-contrastive estimation (NCE) [26, 27, 28, 29], which enables optimization of unnormalized statistical models without explicitly computing the partition function:

$$\max_{\psi,\,\nu} \; \mathbb{E}_{\substack{(s,a)\sim\tau^e,\\ s'\sim P(\cdot|s,a),\; s''_i\sim\rho}} \left[\log \frac{F(s,a,s')}{F(s,a,s') + \Sigma_{i=1}^{k} F(s,a,s''_i)}\right], \tag{8}$$

where $F(s,a,s') = \exp(\psi(s,a)^\top \nu(s'))$, s' denotes a positive sample drawn from the transition distribution $P(\cdot|s,a)$, and s''_i for $i = 1,\ldots,k$ are negative samples from marginalized state distribution $\rho(s) = \int \tau^e(s,a)da$.

Connection to behavioral dynamics. With simple algebra, we obtain the quadratic potential function $P(s'|s,a) \propto q(s') \exp\left(\|\psi(s,a)\|^2/2\right) \exp\left(-\|\psi(s,a) - \nu(s')\|^2/2\right) \left(\|\nu(s')\|^2/2\right)$ from Eq. 7. By assuming ν as an identity map, $Z(s,a)$ as a constant and $q(s') \propto \exp(-||s'||^2/2)$, Eq. 7 reduces to a Gaussian transition model: $P(s'|s,a) = \frac{1}{Z}\exp\left(-\|s' - \psi(s,a)\|^2/2\right)$, which matches standard assumptions in behavioral modeling discussed in Sec. 2. Thus, Eq. 7 provides a more general and flexible framework that extends classical dynamics-based approaches. Moreover, in practice, using Eq. 7 results in a unimodal distribution over s' which closely matches the observed behavior data, while Eq. 2 fails to do so. Thus, while the discrete formulation is theoretically valid in continuous domains, we adopt the parameterization in Eq. 7 for continuous state and action spaces, as it more effectively supports the learning of motif representations underlying animal behavior.

Motif-based policy learning. Since $Z(s,a)$ is intractable, we approximate ϕ via a learned mapping $f : \psi \to \phi$. This yields $Q(s,a) = f\big(\psi(s,a)\big)^\top u$ for later policy learning. However, direct MLE is infeasible due to the continuous action space. This brings us back to the challenge of estimating an unnormalized energy function, $\pi(a|s) \propto \exp(Q(s,a)) \propto \exp(f(\psi(s,a))^\top u)$. Thus, it's reasonable to apply NCE here again,

$$\max_{f,\,u} \; \mathbb{E}_{\substack{(s,a)\sim\tau^e,\\ (s'_i,a'_i)\sim\tau^e}} \left[\log \frac{G(s,a)}{G(s,a) + \Sigma_{i=1}^{k} G(s,a'_i)}\right], \tag{9}$$

where $G(s,a) = \exp(f(\psi(s,a))^\top u)$, $\{s,a\}$ are positive samples and $\{s'_i, a'_i\}$ are negative samples. Identifiability discussions are provided in Appendix P.

4.3 Understanding Animal Behavior via Motifs

In this section, we discuss how motifs ϕ and weights u can be used to describe animal behavior trajectories, by allowing u to vary across tasks or time points and dynamically modulate each motif. We consider two scenarios: (1) discrete state-action spaces in a multi-task setting, and (2) continuous state-action spaces in a time-varying setting. In both settings, $u(t)$ can be viewed as a matrix whose columns correspond to task- or time-specific coefficients over a shared motif basis.

Scenario 1. Consider a mouse navigating a maze, where trajectories are discretized into a finite state and action space. We assume the animal switches among T strategies, each associated with a distinct reward function. We first learn shared motifs $\phi(s,a)$ using Eq. 5, and then fit task-specific policies $\pi(a|s,t)$ via Eq. 6. This yields T sets of coefficients $u(t)$, one per task, all defined over a common motif basis. This formulation captures behavioral variation across tasks as changes in motif weights while preserving shared structure.

Scenario 1 mainly illustrates how to capture shared structure across multiple discrete behavioral strategies through compositional reuse of motifs. However, many natural behaviors are not organized as discrete task switches, but instead evolve continuously over time with gradual transitions and long-range temporal dependencies. To capture these properties, we next consider a continuous state-action setting with time-varying motif weights.

Scenario 2. We now consider a more naturalistic setting in which a mouse freely moves in a cage, with states represented by continuous body-part coordinates and actions represented by velocities. We learn $\psi(s,a)$ using Eq. 8, and estimate f and $u(t)$ via Eq. 9. This results in motifs $\phi(s,a) = f(\psi(s,a))$, and policies constructed as time-varying combinations of these motifs through $u(t)$.

In this scenario, the policy is represented by a generalized linear combination of basis motifs (Eq. 9), with two key properties: (1) **Continuity.** Unlike models with abrupt discrete switches, our method has motif weights that evolve smoothly over time, capturing gradual behavioral changes. (2) **Compositionality.** Unlike models assigning one discrete label to one timestep, our method has

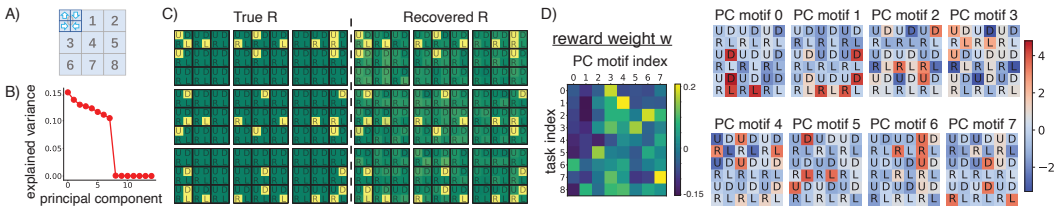

Figure 2: **A.** State-action map: each grid contains four action values (Up, Down, Left, Right). In task i, rewards are assigned to actions moving toward the ith location. **B.** Explained variance of the top 15 principal components; variance drops near zero after PC7. **C.** Left: ground-truth rewards for all 9 tasks (yellow indicates high reward and green indicates low reward). Right: recovered rewards. **D.** Task-specific reward weights w and the top 8 principal-component motifs derived from ϕ.

multiple motifs that can be active simultaneously, allowing behaviors to be expressed as combinations of shared components.

Taken together, MCD provides a flexible and interpretable representation of behavioral dynamics as continuous compositions of shared motifs. In Sec. 5, we demonstrate (1) what motifs we have learned, and (2) how they are combined to generate behavior across simulated and real datasets.

4.4 Reward Recovery

Given the learned motif weights $u(t)$, we can further recover reward parameters $w(t)$ in $r(s, a, t) = \phi(s, a)^\top w(t)$ via

$$w(t) = u(t) - \gamma \int V(s', t)\, \mu(s')\, q(s')\, \mathrm{d}s', \tag{10}$$

where $V(s, t) = \log \sum_a \exp Q(s, a, t)$. This allows us to recover the task-specific or time-varying internal reward function $r(s, a, t)$ used by animals. This formulation aligns with inverse reinforcement learning (IRL) [30]. In the context of animal behavior [31, 32, 33], recovering such rewards offers insight into internal driving factors for decision-making processes. Since $V(s, t)$ is only tractable in discrete settings, we validate this approach by visualizing inferred rewards in the discrete experiments.

5 Experiments

We organize the experiments as a progression from controlled validation to real continuous animal behavior. We first study simulated environments with known reward structure, including a multi-task gridworld and a continuous Lunar Lander setting (Appendix M), to evaluate whether MCD can recover interpretable motifs and reward structure. We then apply MCD to real mouse behavior, first in maze navigation and finally in freely moving natural behavior. Together, these experiments evaluate the practical applicability of MCD across discrete and continuous domains.

5.1 Application to simulated multi-task gridworld

The environment consists of a 3×3 gridworld with four actions per state: Up, Down, Left, and Right (Fig. 2A). In task $t \in \{0, \ldots, 8\}$, rewards are assigned to (s, a) pairs that move toward target state t. Fig. 2C (left) shows the ground-truth reward for all tasks. For each episode, the agent starts from a random state and navigates toward the target. Additional details are provided in Appendix H.

Following Sec. 4.1, we learn a shared motif representation $\phi(s, a)$ and use it to parameterize task-specific policies $\pi(a|s, t)$. Since computational complexity scales linearly with the motif dimension, we use a relatively large representation size ($d = 64$) to maximize coverage of the motif space. Using the learned representation, we recover the reward function $r(s, a, t) = \phi(s, a)^\top w(t)$, as described in Sec. 4.4. The recovered rewards closely match the ground truth (Fig. 2C, right), achieving a Pearson correlation of 0.96. This suggests that the learned motifs are sufficient to reconstruct task structure from behavior alone. Visualizations of these motifs are shown in the Appendix H. Discussion of motif dimension is provided in Appendix S.1.

To interpret the learned representation, we apply principal component analysis (PCA) to the motif matrix ϕ. Only eight principal components explain most of the variance (Fig. 2B), indicating that the effective motif space is low-dimensional. We visualize the top principal-component motifs and their corresponding task-specific coefficients $w(t)$ in Fig. 2D. For example, motif 0 assigns positive

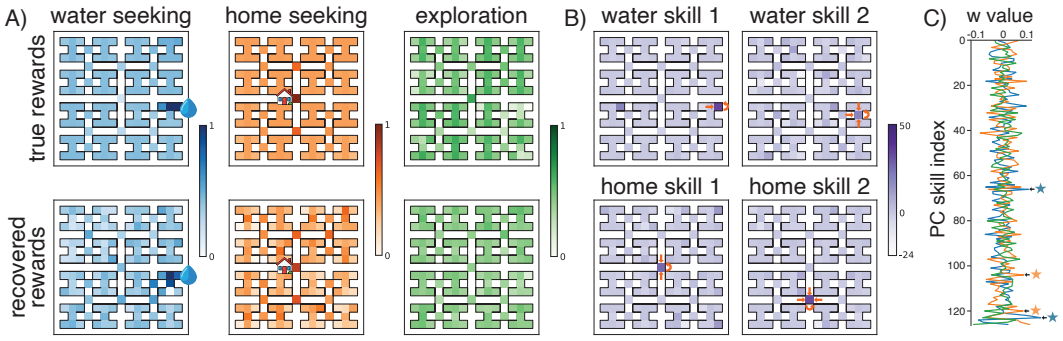

Figure 3: The maze contains 63 T-junctions and 64 terminal nodes. The central root is connected to the home cage. The water port is placed at one terminal node. At each T-junction, an action denotes a transition to one of the adjacent nodes (the parent or the left/right child). The house and droplet icons indicate the home cage and the water port, respectively. **A.** Ground-truth and recovered rewards for the three behavioral modes. Darker colors denote higher reward. **B.** Dominant motifs for the water-seeking and home-seeking behaviors. Orange arrows denote high-value motifs. Darker colors denote higher state values $v(s)$ defined in the main text. **C.** Task-specific coefficients w for all motifs (blue, orange, and green indicate water seeking, home seeking, and exploration, following **A**). Positive coefficients w indicate a motif contributes highly to that recovered reward. The two blue stars highlight the two water-seeking motifs in **B**, and orange stars highlight home-seeking motifs in **B**.

values to transitions toward the bottom-left region and negative values to transitions toward the middle-right region, corresponding to movement from State 5 toward the State 6. Consistent with this interpretation, the corresponding motif weight in w is strongly positive for Task 6 and negative for Task 5. Similar structure appears across other motifs and tasks, demonstrating that the learned motifs provide interpretable building blocks for reward composition. We further evaluate MCD in a simulated continuous-control task Lunar Lander in Appendix M, demonstrating that motif discovery can generalize to the continuous case.

5.2 Application to animal navigation behavior

Dataset and model setup. We evaluate our method on a real discrete animal behavior dataset from [34]. In this experiment, a thirsty mouse is trained to start from home and navigate to the water port in a binary-tree maze (Fig. 3A). The states correspond to the mouse's location in the maze. The actions correspond to transitions to the left/right parent or child node. Although the dominant behavioral drive is water seeking, the mouse also explores unfamiliar regions and returns to the home cage for shelter. These competing motivations cannot be adequately captured by a single reward function, making this dataset well-suited for studying reusable motifs underlying multiple behavioral strategies.

We first apply the segmentation algorithm from [31] to partition trajectories into three interpretable behavioral modes: water seeking, home seeking, and exploration, and apply MCD to infer the corresponding reward functions (Fig. 3A, top row). Water seeking denotes directed movement toward the water port, home seeking toward the maze exit, and exploration contains the remaining non-directed trajectories. While the true internal rewards are unknown, these inferred rewards generate policies that closely match observed behavior and therefore serve as effective ground truth. Using the discrete formulation in Sec. 4.1, we learn a shared motif representation across all three tasks. We set the motif dimension to $d = 127$ to maximize coverage of the motif space (Appendix S.1).

Results. The recovered task-specific rewards capture the most prominent part of inferred ground truth (Fig. 3A, bottom row). In the water-seeking condition, rewards (dark areas) peak near the water port and along trajectories leading toward it. In the home-seeking condition, the dominant reward concentrates around the home cage. Exploration produces a comparatively uniform reward landscape, with reduced value near the water port, suggesting temporary suppression of water-seeking behavior. We also construct a "perfect" reward map where (1) the water-seeking reward function is one-hot at the water port, (2) the home-seeking reward function is one-hot at the home cage, and (3) the exploration reward function has a uniform small reward everywhere. We calculate the correlation of our recovered map and ground truth with the perfect map, and find their correlations are similar (0.4). Therefore, our recovered reward map matches the prominent part of the ground truth.

To interpret the learned representation, we apply PCA to the motif matrix and visualize representative principal-component motifs. Fig. 3B shows selected motifs with the largest task-specific coefficients w (top row: water-seeking, bottom row: home-seeking) in Fig. 3C. For each selected motif $\phi_i(s, a)$,

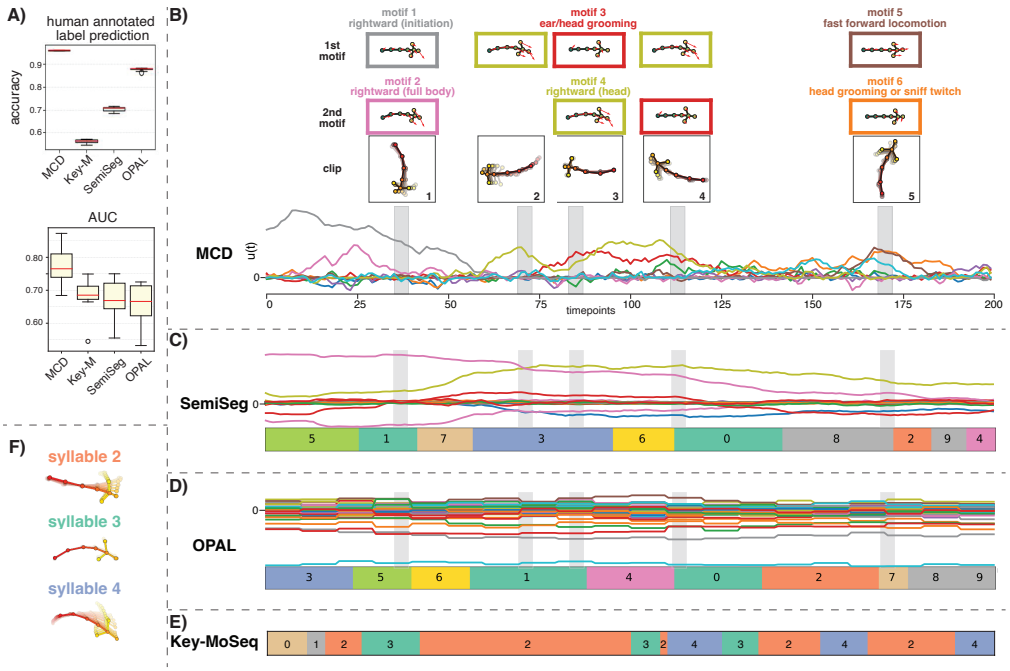

Figure 4: **A.** Boxplot for human-annotated label prediction accuracy (top) and trajectory discrimination AUC (bottom). AUC is evaluated on 10 sequences (length=100) with 1000 negative samples per sequence. **B.** Example motif weights $u(t)$ and representative motifs inferred by MCD. The middle "clip" row shows real animal behavior clips. **C-D.** Latent representations and segmentation results for SemiSeg and OPAL. **E-F.** Segmentation results and representative syllables from Keypoint-MoSeq [2].

we scan the whole state-action space and pick those (s, a) pairs with the highest motif values (shown with orange arrows). Purple shading indicates the state value $v_i(s')$, which is the average motif value pointing to this state: $v_i(s') = \sum_{\{(s,a)|P(s'|s,a)=1\}} \phi_i(s, a)$.

Water-related motifs promote transitions toward the water port, whereas home-related motifs favor movement toward the cage. Interestingly, most motifs are dominated by a single behavioral mode rather than shared uniformly across tasks (Fig. 3C), reflecting competition between underlying motivations. Details for these motifs are provided in Appendix I. Unlike previous reward-discovery approaches on this dataset that lack motifs [33, 31], MCD learns a shared motif basis that simultaneously explains multiple behavioral strategies through different motif combinations.

5.3 Application to animal free-moving behavior

Dataset and model setup. To evaluate MCD on continuous natural behavior, we use a free-moving mouse dataset from [2]. States correspond to the (x, y) coordinates of eight body parts, including the head, nose, ears, and spine nodes. Actions are defined as body-part velocities, $a_t = (s_{t+1} - s_t)/\delta t$, with $\delta t = 1$ ($1/30$s in real time). This dataset contains rich mixtures of grooming, locomotion, and postural adjustment behaviors, making it suitable for testing whether complex movements can be represented through a compact set of reusable motifs.

We use the continuous formulation in Sec. 4.2 to learn motif representations $\phi(s, a)$ and time-varying weights $u(t)$ (Sec. 4.3). To encourage temporal smoothness, we place a Gaussian random-walk prior on $u(t)$: $u(t) \sim \mathcal{N}(u(t-1), \sigma^2 I)$. We optimize f and $u(t)$ via coordinate descent. Since our goal is to analyze behavioral structure rather than recover rewards, we focus on interpreting ϕ and $u(t)$ rather than performing IRL. Performance saturates beyond motif dimension $d = 64$ (Appendix S.1), so we use $d = 64$ in practice. Additional training details are provided in Appendix J.

We compare MCD against three representative baselines (Sec. 2): Keypoint-MoSeq [2], an HMM-based behavior segmentation method; SemiSeg [13], a clustering-based approach; and OPAL [18], an RL-based skill learning framework.

Results. To evaluate whether the inferred latent weights $u(t)$ capture meaningful behavioral structure, we train a two-layer neural network to decode human-annotated labels. MCD achieves the highest

test accuracy (Fig. 4A top, $p < 0.05$), suggesting that its motif representations align closely with behavioral dynamics. Additional details are provided in Appendix N.

We further evaluate each model's ability to distinguish real trajectories from mismatched state-action pairs using the area under the receiver operating characteristic curve (AUC). This method allows direct comparison between our unnormalized energy function and other models' normalized likelihood. Given $(s, a), (s', a') \sim \tau^e$, positive samples are true pairs (s, a), while negative samples are mismatch pairs (s, a'). With respect to the prediction score for AUC, we use action log-likelihood for Keypoint-MoSeq and OPAL, Gaussian log-likelihood for SemiSeg, and negative energy $\phi(s, a)^\top u(t)$ for MCD. MCD achieves the highest AUC across all baselines (paired t-test, $p < 0.05$; Fig. 4A bottom), indicating the strongest ability to capture real dynamics.

Beyond quantitative evaluation, we also qualitatively visualize the learned motifs and their temporal activations on an example behavior sequence of length=250 (Fig. 4B bottom). We extract five short clips ((Fig. 4 B, the third row), each associated with a distinct motif combination (Fig. 4 B top two rows). We visualize the dominant motifs for each clip, and show their average poses and actions, computed by averaging (s, a) that most strongly activate that motif. Additional visualizations are provided in Appendix J. For comparison, we visualize latent trajectories from SemiSeg and OPAL (Fig. 4C-D), together with Keypoint-MoSeq segmentations (Fig. 4E). The segmentations of SemiSeg and OPAL are calculated through KMeans ($k = 10$) on their first 10 PCs of the latents.

MCD captures both transitions and mixtures of behaviors through smoothly varying motif activations. In Clip 1, right-turning behavior is represented by the joint activation of two turning motifs, with motif 1 emphasizing stronger head motion. Clips 2–4 exhibit alternating turning and grooming dynamics, reflected by alternating motif dominance of motif 3 and motif 4. Clip 5 combines forward locomotion and sniffing, captured through simultaneous activation of two motifs. Notably, motif 4 reappears across multiple clips, indicating that motifs are reusable components rather than clip-specific states.

In contrast, baseline methods show less consistent structure. For example, in Clip 2-4, SemiSeg assigns nearly identical latent states to qualitatively different behaviors. OPAL fails to produce interpretable and dominant motifs for interpretation. Both models lack recurring motifs, suggesting their latents can not decompose behaviors into a simple subset. Keypoint-MoSeq often assigns distinct behaviors to the same syllable. For example, Clip 2 and 3 receive the same label despite corresponding to turning and grooming behaviors respectively. Moreover, mixed behaviors in Clip 5 with simultaneous locomotion and sniffing are not represented compositionally in Keypoint-MoSeq.

Taken together, these results show that MCD captures animal behavior through a compact and reusable set of motifs whose continuous combinations explain complex behavioral dynamics. Additional ablations on motif dimension, policy noise distribution, dataset partitioning, Gaussian random walk priors and temporal resolution are provided in Appendix F. We further assess generative quality through rollout experiments in Appendix T. To show that MCD achieves more than just a smoothed discrete model, we carry out an ablation study in Appendix R.

5.4 Application to macaque character drawing

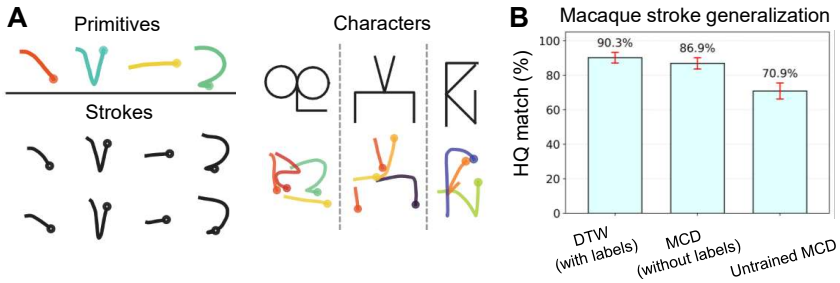

Figure 5: **A.** The strokes can be categorized into different primitives, and the primitives are reused in different characters. **B.** Primitive HQ coverage for the test set. Bars show HQ rates. Red error bars show 95% bootstrap intervals. All methods use the same evaluation strokes.

We evaluate MCD on the macaque drawing dataset of Tian et al. [35], in which monkeys copy geometric figures on a touchscreen. After practising simple shapes, the animals were presented with novel, more complex figures. The original study found that they reused familiar drawing strokes,

or *primitives*, when producing these novel characters (Fig. 5A). We investigate whether MCD can identify the same reusable primitives in previously unseen drawings. We train MCD on isolated primitives and practised characters, and evaluate it on held-out novel characters.

Here, the state is defined as the 2D drawing position, and the action is the velocity. Different tasks correspond to different characters. Since each character consists of multiple strokes, and each stroke is a sequence of motifs, we first represent each stroke r by averaging its motif features $\phi(s_{r,t}, a_{r,t})$ over its T_r time steps: $z_r = \frac{1}{T_r}\sum_{t=1}^{T_r}\phi(s_{r,t}, a_{r,t})$. We then run K-means on isolated primitives to obtain $K = 19$ clusters and their centroids. The discovered clusters achieve 89.8% Hungarian alignment (chance level=14.7%) with the predefined labels.

To quantify primitive reuse in the novel characters, we measure whether these strokes in the novel characters can match the clusters constructed from the training dataset. Each test stroke representation z_r is compared with all 19 primitive centroids using cosine-based similarity $S(z_r, \mu_c) = 1/\left(2 - \cos\langle z_r, \mu_c\rangle\right)$ and assigned to its nearest primitive, $\hat{c}_r = \arg\max_c S(z_r, \mu_c)$. The stroke is considered a high-quality (HQ) match if the similarity surpasses the threshold τ_c. We set τ_c to the 2.5th percentile of the similarities within each assigned group (Appendix K). The HQ rate is then computed by averaging the fraction of HQ strokes within characters in test dataset. MCD achieves 86.9% without labels, close to the 90.3% achieved by the original label-based dynamic time warping (DTW) reference and substantially above untrained MCD (Fig. 5B). These results indicate that MCD can discover useful primitives and recognize their reuse in novel character compositions.

6 Discussion

In this work, we introduced *Motif-based Continuous Dynamics discovery (MCD)*, a reinforcement-learning-based framework for understanding animal behavior through reusable motor motifs and their continuous compositions. Unlike traditional behavior segmentation approaches that model behavior as transitions between discrete syllables, MCD represents behavior as a combination of shared motifs learned directly from transition dynamics. Across simulated and real animal behavior datasets, we showed that MCD can recover interpretable motif structures, reconstruct reward functions, and provide more faithful representations of continuous pose dynamics than existing segmentation methods. Together, these results suggest that natural behavior is better characterized as continuous compositions of reusable motifs rather than abrupt switches between discrete behavioral states.

Our framework is motivated by longstanding ideas in neuroscience and motor control that complex movements arise from combinations of a finite set of reusable motor primitives [7, 36, 37, 38, 39]. MCD operationalizes this perspective within a reinforcement learning framework by treating motifs as low-level basis components shared across policies and rewards. Importantly, the learned motifs emerge directly from behavioral dynamics without requiring neural recordings, reward annotations, or hand-designed behavioral labels.

MCD also provides a principled framework for relating continuous behavioral dynamics to low-dimensional neural representations in future studies. The learned motifs $\phi(s, a)$ define low-level movement components that may serve as behavioral regressors for testing whether neural populations encode specific motor primitives. Meanwhile, the time (task)-varying policy weights and reward-related coefficients characterize how motifs are combined and modulated over time (task), potentially relating to control- and value-related neural signals.

Beyond neuroscience, MCD is also relevant for studying naturalistic behavior in ethology. By allowing motifs to co-occur and evolve continuously, the framework naturally captures multi-scale behavioral organization (Appendix Q) without imposing discrete state assumptions. This makes MCD suitable for studying exploratory behavior, foraging, grooming, and other unconstrained behaviors that unfold across multiple timescales.

Several limitations remain for future work. First, the quality of the learned motifs depends on the quality of behavioral recordings, and tracking noise or pose estimation errors may degrade performance. Second, while the current framework focuses solely on behavior, incorporating multimodal information such as neural activity, sensory inputs, or environmental context may further improve motif discovery and interpretability. Finally, although MCD provides interpretable behavioral motifs, establishing direct correspondence between these motifs and specific neural dynamics still requires experimental validation.

Acknowledgement

This work was supported by NSF CAREER Award IIS-2540614.

References

[1] Alexander B Wiltschko, Matthew J Johnson, Giuliano Iurilli, Ralph E Peterson, Jesse M Katon, Stan L Pashkovski, Victoria E Abraira, Ryan P Adams, and Sandeep Robert Datta. Mapping sub-second structure in mouse behavior. *Neuron*, 88(6):1121–1135, 2015.

[2] Caleb Weinreb, Jonah E Pearl, Sherry Lin, Mohammed Abdal Monium Osman, Libby Zhang, Sidharth Annapragada, Eli Conlin, Red Hoffmann, Sofia Makowska, Winthrop F Gillis, et al. Keypoint-moseq: parsing behavior by linking point tracking to pose dynamics. *Nature Methods*, 21(7):1329–1339, 2024.

[3] Kevin Luxem, Petra Mocellin, Falko Fuhrmann, Johannes Kürsch, Stephanie R Miller, Jorge J Palop, Stefan Remy, and Pavol Bauer. Identifying behavioral structure from deep variational embeddings of animal motion. *Communications Biology*, 5(1):1267, 2022.

[4] Alexander I Hsu and Eric A Yttri. B-soid, an open-source unsupervised algorithm for identification and fast prediction of behaviors. *Nature communications*, 12(1):5188, 2021.

[5] Gordon J Berman, Daniel M Choi, William Bialek, and Joshua W Shaevitz. Mapping the stereotyped behaviour of freely moving fruit flies. *Journal of The Royal Society Interface*, 11(99):20140672, 2014.

[6] Alessandro Santuz, Turgay Akay, William P Mayer, Tyler L Wells, Arno Schroll, and Adamantios Arampatzis. Modular organization of murine locomotor pattern in the presence and absence of sensory feedback from muscle spindles. *The Journal of physiology*, 597(12):3147–3165, 2019.

[7] Tamar Flash and Binyamin Hochner. Motor primitives in vertebrates and invertebrates. *Current opinion in neurobiology*, 15(6):660–666, 2005.

[8] Bo Dai, Niao He, Yunpeng Pan, Byron Boots, and Le Song. Learning from conditional distributions via dual embeddings. In *Artificial intelligence and statistics*, pages 1458–1467. PMLR, 2017.

[9] Tongzheng Ren, Tianjun Zhang, Lisa Lee, Joseph E Gonzalez, Dale Schuurmans, and Bo Dai. Spectral decomposition representation for reinforcement learning. *arXiv preprint arXiv:2208.09515*, 2022.

[10] Dmitry Shribak, Chen-Xiao Gao, Yitong Li, Chenjun Xiao, and Bo Dai. Diffusion spectral representation for reinforcement learning. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[11] Markus Marks, Qiuhan Jin, Oliver Sturman, Lukas von Ziegler, Sepp Kollmorgen, Wolfger von der Behrens, Valerio Mante, Johannes Bohacek, and Mehmet Fatih Yanik. Deep-learning-based identification, tracking, pose estimation and behaviour classification of interacting primates and mice in complex environments. *Nature machine intelligence*, 4(4):331–340, 2022.

[12] Cristina Segalin, Jalani Williams, Tomomi Karigo, May Hui, Moriel Zelikowsky, Jennifer J Sun, Pietro Perona, David J Anderson, and Ann Kennedy. The mouse action recognition system (mars) software pipeline for automated analysis of social behaviors in mice. *Elife*, 10:e63720, 2021.

[13] Matthew R Whiteway, Evan S Schaffer, Anqi Wu, E Kelly Buchanan, Omer F Onder, Neeli Mishra, and Liam Paninski. Semi-supervised sequence modeling for improved behavioral segmentation. *bioRxiv*, pages 2021–06, 2021.

[14] Julia Costacurta, Lea Duncker, Blue Sheffer, Winthrop Gillis, Caleb Weinreb, Jeffrey Markowitz, Sandeep R Datta, Alex Williams, and Scott Linderman. Distinguishing discrete and continuous behavioral variability using warped autoregressive hmms. *Advances in neural information processing systems*, 35:23838–23850, 2022.

[15] Diego Aldarondo, Josh Merel, Jesse D Marshall, Leonard Hasenclever, Ugne Klibaite, Amanda Gellis, Yuval Tassa, Greg Wayne, Matthew Botvinick, and Bence P Ölveczky. A virtual rodent predicts the structure of neural activity across behaviours. *Nature*, 632(8025):594–602, 2024.

[16] Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. In *International Conference on Learning Representations*, 2019.

[17] Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. In *International Conference on Learning Representations*, 2020.

[18] Anurag Ajay, Aviral Kumar, Pulkit Agrawal, Sergey Levine, and Ofir Nachum. Opal: Offline primitive discovery for accelerating offline reinforcement learning. In *International Conference on Learning Representations*, 2021.

[19] Karl Pertsch, Youngwoon Lee, and Joseph Lim. Accelerating reinforcement learning with learned skill priors. In *Conference on robot learning*, pages 188–204. PMLR, 2021.

[20] Tianjun Zhang, Tongzheng Ren, Mengjiao Yang, Joseph Gonzalez, Dale Schuurmans, and Bo Dai. Making linear mdps practical via contrastive representation learning. In *International Conference on Machine Learning*, pages 26447–26466. PMLR, 2022.

[21] Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *Proceedings of the 35th International Conference on Machine Learning (ICML)*, 2018.

[22] Gene H Golub and Christian Reinsch. Singular value decomposition and least squares solutions. In *Handbook for automatic computation: volume II: linear algebra*, pages 134–151. Springer, 1971.

[23] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

[24] Lloyd N Trefethen and David Bau. *Numerical linear algebra*. SIAM, 2022.

[25] Jeff Z HaoChen, Colin Wei, Adrien Gaidon, and Tengyu Ma. Provable guarantees for self-supervised deep learning with spectral contrastive loss. *Advances in neural information processing systems*, 34:5000–5011, 2021.

[26] Ga Wu, Maksims Volkovs, Chee Loong Soon, Scott Sanner, and Himanshu Rai. Noise contrastive estimation for scalable linear models for one-class collaborative filtering. *arXiv preprint arXiv:1811.00697*, 2018.

[27] Zhuang Ma and Michael Collins. Noise contrastive estimation and negative sampling for conditional models: Consistency and statistical efficiency. *arXiv preprint arXiv:1809.01812*, 2018.

[28] Michael Gutmann and Aapo Hyvärinen. Noise-contrastive estimation: A new estimation principle for unnormalized statistical models. In *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics (AISTATS)*, pages 297–304, 2010.

[29] Michael U Gutmann and Aapo Hyvärinen. Noise-contrastive estimation of unnormalized statistical models, with applications to natural image statistics. *Journal of Machine Learning Research*, 13(Feb):307–361, 2012.

[30] Brian D Ziebart, Andrew L Maas, J Andrew Bagnell, Anind K Dey, et al. Maximum entropy inverse reinforcement learning. In *Aaai*, volume 8, pages 1433–1438. Chicago, IL, USA, 2008.

[31] Jingyang Ke, Feiyang Wu, Jiyi Wang, Jeffrey Markowitz, and Anqi Wu. Inverse reinforcement learning with switching rewards and history dependency for characterizing animal behaviors. In *Forty-second International Conference on Machine Learning*, 2025.

[32] Hao Zhu, Brice De La Crompe, Gabriel Kalweit, Artur Schneider, Maria Kalweit, Ilka Diester, and Joschka Boedecker. Multi-intention inverse q-learning for interpretable behavior representation. *arXiv preprint arXiv:2311.13870*, 2023.

[33] Zoe Ashwood, Aditi Jha, and Jonathan W Pillow. Dynamic inverse reinforcement learning for characterizing animal behavior. *Advances in neural information processing systems*, 35:29663–29676, 2022.

[34] Matthew Rosenberg, Tony Zhang, Pietro Perona, and Markus Meister. Mice in a labyrinth show rapid learning, sudden insight, and efficient exploration. *Elife*, 10:e66175, 2021.

[35] Lucas Y Tian, Kedar Garzón Gupta, Daniel J Hanuska, Adam G Rouse, Mark AG Eldridge, Marc H Schieber, Xiao-Jing Wang, Joshua B Tenenbaum, and Winrich A Freiwald. Neural representation of action symbols in primate frontal cortex. *Nature*, pages 1–11, 2026.

[36] Emilio Bizzi, Simon F Giszter, Eric Loeb, Fernando A Mussa-Ivaldi, and Philippe Saltiel. Modular organization of motor behavior in the frog's spinal cord. *Trends in neurosciences*, 18(10):442–446, 1995.

[37] Andrea d'Avella, Philippe Saltiel, and Emilio Bizzi. Combinations of muscle synergies in the construction of a natural motor behavior. *Nature neuroscience*, 6(3):300–308, 2003.

[38] Matthew C Tresch, Philippe Saltiel, and Emilio Bizzi. The construction of movement by the spinal cord. *Nature neuroscience*, 2(2):162–167, 1999.

[39] Corey B Hart and Simon F Giszter. A neural basis for motor primitives in the spinal cord. *Journal of Neuroscience*, 30(4):1322–1336, 2010.

[40] Ali Rahimi and Benjamin Recht. Random features for large-scale kernel machines. In *Advances in neural information processing systems (NeurIPS)*, volume 20, pages 1177–1184, 2007.

[41] Divyansh Garg, Shuvam Chakraborty, Chris Cundy, Jiaming Song, and Stefano Ermon. Iq-learn: Inverse soft-q learning for imitation. *Advances in Neural Information Processing Systems*, 34:4028–4039, 2021.

[42] Alexandros Paraschos, Christian Daniel, Jan Peters, and Gerhard Neumann. Probabilistic movement primitives. In *Advances in Neural Information Processing Systems*, volume 26, 2013.

[43] Matteo Saveriano, Fares J Abu-Dakka, Aljaž Kramberger, and Luka Peternel. Dynamic movement primitives in robotics: A tutorial survey. *The International Journal of Robotics Research*, 42(13):1133–1184, 2023.

[44] Rudolf Lioutikov, Gerhard Neumann, Guilherme Maeda, and Jan Peters. Learning movement primitive libraries through probabilistic segmentation. *The International Journal of Robotics Research*, 36(8):879–894, 2017.

[45] Ilya Kostrikov, Ofir Nachum, and Jonathan Tompson. Imitation learning via off-policy distribution matching. *arXiv preprint arXiv:1912.05032*, 2019.

[46] Rushuai Yang, Chenjia Bai, Hongyi Guo, Siyuan Li, Bin Zhao, Zhen Wang, Peng Liu, and Xuelong Li. Behavior contrastive learning for unsupervised skill discovery. In *International conference on machine learning*, pages 39183–39204. PMLR, 2023.

[47] Yunzhu Li, Jiaming Song, and Stefano Ermon. Infogail: Interpretable imitation learning from visual demonstrations. *Advances in neural information processing systems*, 30, 2017.

[48] Arjun Sharma, Mohit Sharma, Nicholas Rhinehart, and Kris M Kitani. Directed-info gail: Learning hierarchical policies from unsegmented demonstrations using directed information. *arXiv preprint arXiv:1810.01266*, 2018.

[49] Yuxuan Kuang, Haoran Geng, Amine Elhafsi, Tan-Dzung Do, Pieter Abbeel, Jitendra Malik, Marco Pavone, and Yue Wang. Skillblender: Towards versatile humanoid whole-body loco-manipulation via skill blending. *arXiv preprint arXiv:2506.09366*, 2025.

[50] Xue Bin Peng, Yunrong Guo, Lina Halper, Sergey Levine, and Sanja Fidler. Ase: Large-scale reusable adversarial skill embeddings for physically simulated characters. *ACM Transactions On Graphics (TOG)*, 41(4):1–17, 2022.

[51] Karl Pertsch, Youngwoon Lee, Yue Wu, and Joseph J Lim. Demonstration-guided reinforcement learning with learned skills. In *5th Annual Conference on Robot Learning*, 2021.

[52] Avi Singh, Huihan Liu, Gaoyue Zhou, Albert Yu, Nicholas Rhinehart, and Sergey Levine. Parrot: Data-driven behavioral priors for reinforcement learning. In *International Conference on Learning Representations*, 2021.

[53] Seohong Park, Tobias Kreiman, and Sergey Levine. Foundation policies with hilbert representations. In *Forty-first International Conference on Machine Learning*, 2024.

[54] Ahmed Touati and Yann Ollivier. Learning one representation to optimize all rewards. *Advances in Neural Information Processing Systems*, 34:13–23, 2021.

[55] Diana Borsa, Andre Barreto, John Quan, Daniel J. Mankowitz, Hado van Hasselt, Remi Munos, David Silver, and Tom Schaul. Universal successor features approximators. In *International Conference on Learning Representations*, 2019.

[56] Boyuan Chen, Chuning Zhu, Pulkit Agrawal, Kaiqing Zhang, and Abhishek Gupta. Self-supervised reinforcement learning that transfers using random features. *Advances in Neural Information Processing Systems*, 36:56411–56436, 2023.

[57] Mengjiao Yang, Sergey Levine, and Ofir Nachum. TRAIL: Near-optimal imitation learning with suboptimal data. In *International Conference on Learning Representations*, 2022.

[58] Hiroshi Makino. Arithmetic value representation for hierarchical behavior composition. *Nature neuroscience*, 26(1):140–149, 2023.

A Impact statement

Beyond advancing animal behavior research, MCD has broader implications. Positively, a better understanding of motor control mechanisms could, for instance, inform new treatments for movement disorders or inspire more adaptable AI. On the other side, extending these principles to model human behavior carries ethical risks, such as perpetuating or amplifying societal biases present in training data. A robust ethical framework is essential to mitigate such risks in the development and application of these technologies.

B Reproducibility statement

We have taken several steps to ensure the reproducibility of our results. All source code for model training and evaluation is included in the supplementary material, allowing independent verification and replication of our experiments. The complete set of hyperparameter values is documented in the appendix. Additionally, the data preprocessing procedures and evaluation protocols are described in the main text. These resources provide sufficient information for reproducing the results reported in this paper.

C LLM usage

In preparing this manuscript, we employed a large language model (OpenAI ChatGPT, GPT-5) as a writing assistant. The model was used exclusively for polishing English grammar, improving clarity, and suggesting more natural phrasing in certain sections of the text. All scientific content, experimental design, analyses, and interpretations were conceived, written, and verified by the authors. The LLM was not used to generate original research ideas, analyses, or results. To ensure accuracy, all model-suggested edits were carefully reviewed and, where necessary, modified by the authors.

D Licenses

All datasets, environments, and baseline implementations used in this work are publicly available and are properly credited through citations in the main text and references. We use these assets in accordance with their respective licenses and terms of use.

Gymnasium / LunarLander environment. The LunarLander environment from Gymnasium/OpenAI Gym is distributed under the MIT License, with copyright attributed to OpenAI and the Farama Foundation. Our usage complies with the corresponding open-source license terms. The official project page is: `https://github.com/Farama-Foundation/Gymnasium`

Labyrinth navigation dataset. The labyrinth navigation dataset from [34] is released under the MIT License. We use the dataset and associated software in accordance with the license terms. The official project page is: `https://github.com/markusmeister/Rosenberg-2021-Repository`

Keypoint-MoSeq. The Keypoint-MoSeq software and dataset [2] are distributed under the Harvard University Non-Commercial Research and Academic Use Software License, which permits non-commercial academic research use. Our usage complies with these terms. The official project page is: `https://github.com/dattalab/keypoint-moseq`.

Released assets. The implementation code released with this paper is provided solely for academic research and reproducibility purposes.

E Computation resources and execution time

Hardware. All experiments were conducted on a single workstation equipped with an AMD Ryzen 7 7840H CPU, 16 GB RAM, and an NVIDIA GeForce RTX 4060 Laptop GPU (8 GB VRAM). The GPU driver version is 566.14 with CUDA 12.7 support.

Software. All models were implemented in PyTorch with standard CUDA acceleration. Detailed environment and dependency configurations are provided in the supplementary material.

Training cost. The computational cost varies across tasks:

- Gridworld experiments: $\sim$10 minutes per run
- Labyrinth navigation: $\sim$1 hour per run
- Continuous animal behavior dataset: $\sim$8 hours per run

Reproducibility and scalability. All models are trained from scratch without pretraining. The experiments can be reproduced on a single consumer-grade GPU with comparable memory (e.g., 8 GB VRAM). Additional exploratory or preliminary experiments not included in the paper required modest additional compute but follow similar resource profiles.

F Hyperparameter setting

We train MCD using the following hyperparameters:

General hyperparameters.

- Discount factor: $\gamma = 0.99$
- Number of gradient update steps: 1×10^6
- Batch size: 256

Motif representations. The motif representation $\phi(s, a) \in \mathbb{R}^d$ and $\mu(s') \in \mathbb{R}^d$ were adopted with different motif dimensions d depending on the task:

- Gridworld: $d = 64$
- Animal navigation: $d = 127$
- Animal free-moving: $d = 64$

Model architectures.

- **Discrete version:** ϕ and μ are parameterized by one-hidden-layer neural networks (hidden size $= 512$).
- **Continuous version:**
  - f: no hidden layer
  - ν: one hidden layer (hidden size $= 512$), with one normalization layer to ensure the L2-Norm of the output is 1.
  - ψ: no hidden layer, with one normalization layer to ensure the L2-Norm of the output is 1.
- In both cases, u and w are just two matrices, with each column corresponding to $u(t)$ or $w(t)$ at a specific task or timepoint.
- Activation function: For all networks, if it has at least one hidden layer, then all of its activation functions are Exponential Linear Unit (ELU).

Learning rates.

- **Discrete version:**
  $$\phi : 1 \times 10^{-3}, \mu : 1 \times 10^{-3}, u : 3 \times 10^{-4}, w : 3 \times 10^{-4}.$$
- **Continuous version:**
  $$\psi : 5 \times 10^{-4}, \nu : 5 \times 10^{-4}, f : 3 \times 10^{-4}, u : 3 \times 10^{-4}.$$
  During testing, u and f are further optimized on the new sequence using gradient descent with learning rate 1×10^{-3}.

We train SemiSeg and OPAL using the following hyperparameters:

- Discount factor: $\gamma = 0.99$
- Number of epochs: 1×10^6
- Batch size: 4×250 (4 sequences, each of length=250 because this is an RNN-based inference model)
- Latent dimension: $d = 64$
- Learning rate: 1×10^{-4} (tuned to get better results)

Besides, the following loss coefficients are shared across three models for interpretability results.

- Temporal smoothness Gaussian random walk loss: 10
- Sparsity L1-loss: 0.1

G Approximating energy-based formulation with low-rank spectral decomposition

In this part, we show in detail the connection between the EBM formulation (Eq. 7) and the low-rank spectral decomposition formulation (Eq. 2) of the transition kernel $P(s'|s,a)$.

From Eq. 7, by simple algebra, we obtain the quadratic potential function,

$$\begin{aligned} P(s'|s,a) \propto & q(s') \cdot \exp\left(\|\psi(s,a)\|^2/2\right) \cdot \exp\left(\|\nu(s')\|^2/2\right) \\ & \cdot \exp\left(-\|\psi(s,a)-\nu(s')\|^2/2\right). \end{aligned} \tag{11}$$

The term $\exp\left(-\frac{\|\psi(s,a)-\nu(s')\|^2}{2}\right)$ is the Gaussian kernel, for which we apply the random Fourier feature [8, 40] and obtain the spectral decomposition of Eq. 7 as

$$P(s'|s,a) = \langle \phi_\omega(s,a), \mu_\omega(s')q(s') \rangle_{\mathcal{N}(\omega)}, \tag{12}$$

where $\omega \sim \mathcal{N}(0, I)$ is the frequency in the Fourier domain, and

$$\begin{aligned} \phi_\omega(s,a) = & \exp\left(-i\omega^\top \psi(s,a)\right) \\ & \cdot \exp\left(\|\psi(s,a)\|^2/2 - \log Z(s,a)\right), \end{aligned} \tag{13}$$

$$\mu_\omega(s') = \exp\left(-i\omega^\top \nu(s')\right) \cdot \exp\left(\|\nu(s')\|^2/2\right). \tag{14}$$

Note that Eq. 12 needs infinite ω to calculate the expectation. To connect it to finite dimension $\phi(s,a) \in \mathbb{R}^d, \mu(s') \in \mathbb{R}^d$, we use the Monte-Carlo method to approximate it with finite samples,

$$P(s'|s,a) \approx \frac{1}{M} \sum_{i=1}^{M} \phi_{\omega_i}(s,a)\mu_{\omega_i}(s')q(s'). \tag{15}$$

Introduce vectors $\phi(s,a)$ and $\mu(s')$ such that

$$\phi(s,a) := \frac{1}{\sqrt{M}} \left[\phi_{\omega_1}(s,a), \phi_{\omega_2}(s,a), ..., \phi_{\omega_M}(s,a)\right], \tag{16}$$

$$\mu(s') := \frac{1}{\sqrt{M}} \left[\mu_{\omega_1}(s'), \mu_{\omega_2}(s'), ..., \mu_{\omega_M}(s')\right]. \tag{17}$$

Then it's straightforward to see that,

$$\begin{aligned} \phi(s,a)^\top \mu(s')q(s') &= \frac{1}{M} \sum_{i=1}^{M} (\phi_{\omega_i}(s,a)^\top \mu_{\omega_i}(s')) \\ &\approx P(s'|s,a). \end{aligned} \tag{18}$$

Hence, Eq. 7 can, in principle, yield the motif representation introduced earlier.

H Multi-task gridworld dataset

H.1 Dataset

To generate the dataset, we follow this procedure: 1) Use soft value iteration to compute the ground truth Q-function for each task: $Q(s,a,t) = r(s,a,t) + \log \sum_a \exp V(s,t)$; 2) Use the resulting Q-function to define the policy: $\pi(a|s,t) = \exp(Q(s,a,t))/\sum_{a'} \exp(Q(s,a',t))$ and sample trajectories accordingly.

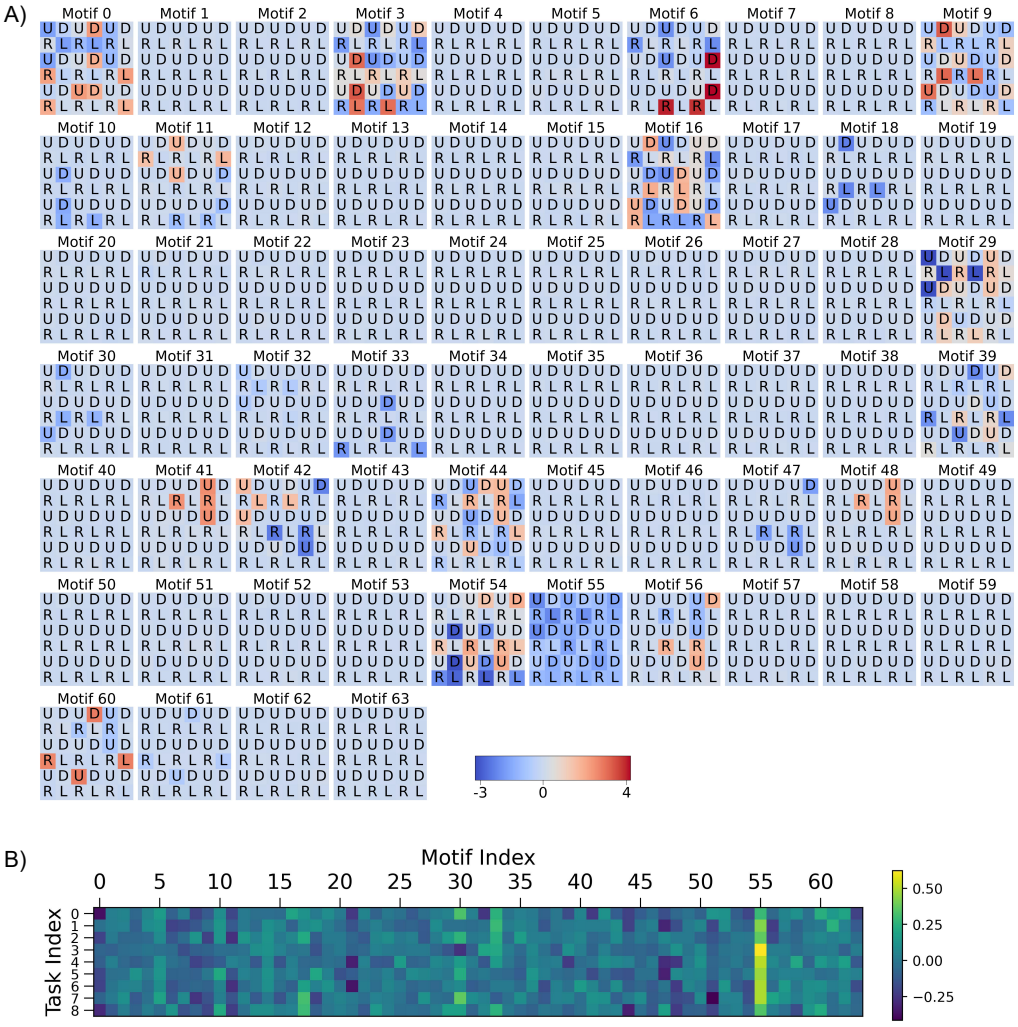

Figure 6: **A.** State-action maps for all 64 motifs. **B.** Reward weight w for all 64 motifs.

H.2 Learned motifs

We visualize all original 64 motifs (Fig. 6) introduced in Sec. 5.1. It shows some meaningful patterns as mentioned before. For example, motif 0 assigns high values to those (s,a) pairs leading to the middle-middle grid and the bottom-middle grid, and assigns low values to the up-left grid and the up-right grid. Thus, it is employed negatively in Task 0 (up-left reward) and Task 2 (up-right reward). However, Task 4 (middle-middle) didn't use this motif and used motif 39 negatively instead. The complex many-to-many relationship between motifs and tasks informs us of the redundancy in the original motifs, which inspires us to use PCA to analyze the principal components of the motif space and simplify the motif weights. It could be seen from the comparison between Fig. 2D and Fig. 6A that principal components are a less redundant description of the motif space.

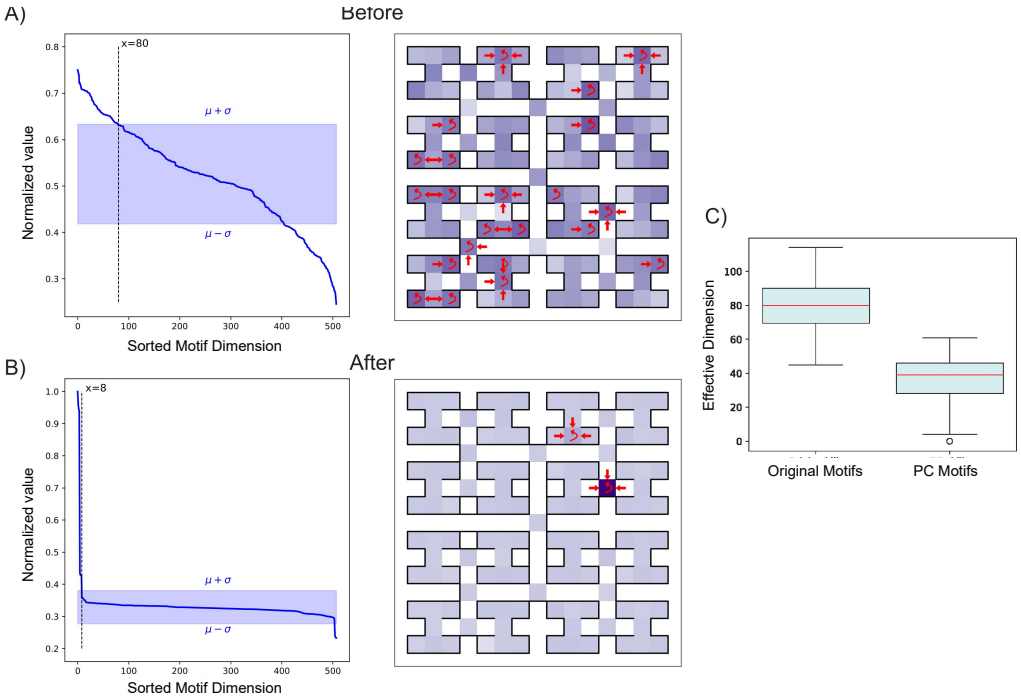

Figure 7: **A.** Left. The value for each (s, a) pair in motif 0 before PCA. Right. The most important (s, a) pairs. **B.** Left. The value for each (s, a) pair in motif 0 after PCA. Right. The most important (s, a) pairs. **C.** Boxplot for the effective dimensions before and after PCA.

I Animal navigation behavior dataset

I.1 Learned motifs

In the original motifs of the labyrinth environment, multiple (s, a) pairs are simultaneously activated, so it is rather hard to analyze which (s, a) pairs are the most important ones that could represent the focus and function of the motif. Given the redundancy of the motif sets, as in Appendix H, we perform PCA to analyze the principle components of the motif space and simplify the motif representation. To show the effect of PCA, we plot one motif (motif 0) before (Fig. 7A right) and after PCA (Fig. 7B right). Basically, we only want to show the most important pairs in one map and do not want low-value pairs to disturb the visualization. To determine how many (s, a) pairs are important, we sort the (s, a) pairs based on the value $\phi(s, a)$ in motif 0, i.e., the first feature of the output of $\phi(s, a)$ (Fig. 7A left and B left). It could be seen straightforwardly that after PCA, the motif becomes more concentrated on several (s, a) pairs. We calculate the mean μ and standard deviation σ across all motifs and all dimensions and take $\mu + \sigma$ as the threshold, above which (s, a) pairs are deemed the most important ones and are shown on the right. We show 80 pairs before PCA and 8 pairs after PCA. The number of the most important pairs in each motif is called the "effective dimension." The effective dimension is calculated across all motifs (Fig. 7C). Paired t-test ($p = 1.3 \times 10^{-51}$) shows that there exists a significant decrease of effective dimensions after PCA. So the map becomes more distinct and functionally separated.

J Animal free-moving behavior dataset

J.1 Dataset and training

We split the full dataset into training and test trajectories in an 8:2 ratio. We first learn both f and $u(t)$ on the training set. Then, given the learned f, we estimate $u(t)$ on the test set. Here, f is a time-invariant model parameter shared across all time, while $u(t)$ is a time-dependent variable that must be inferred separately for each test trajectory and cannot be transferred from training.

J.2 Learned motifs

We show all motifs learned from the 200-timestep video clip of the free-moving mouse mentioned in Sec. 5.3. We have completed the visualization of those motifs that were previously omitted due to their perceived lack of importance. Due to the increased number of displayed motifs, we had to renumber each motif. We show the present motif number above the motif motion field figure, and previous numbers (if applicable) in the parentheses following the present number.

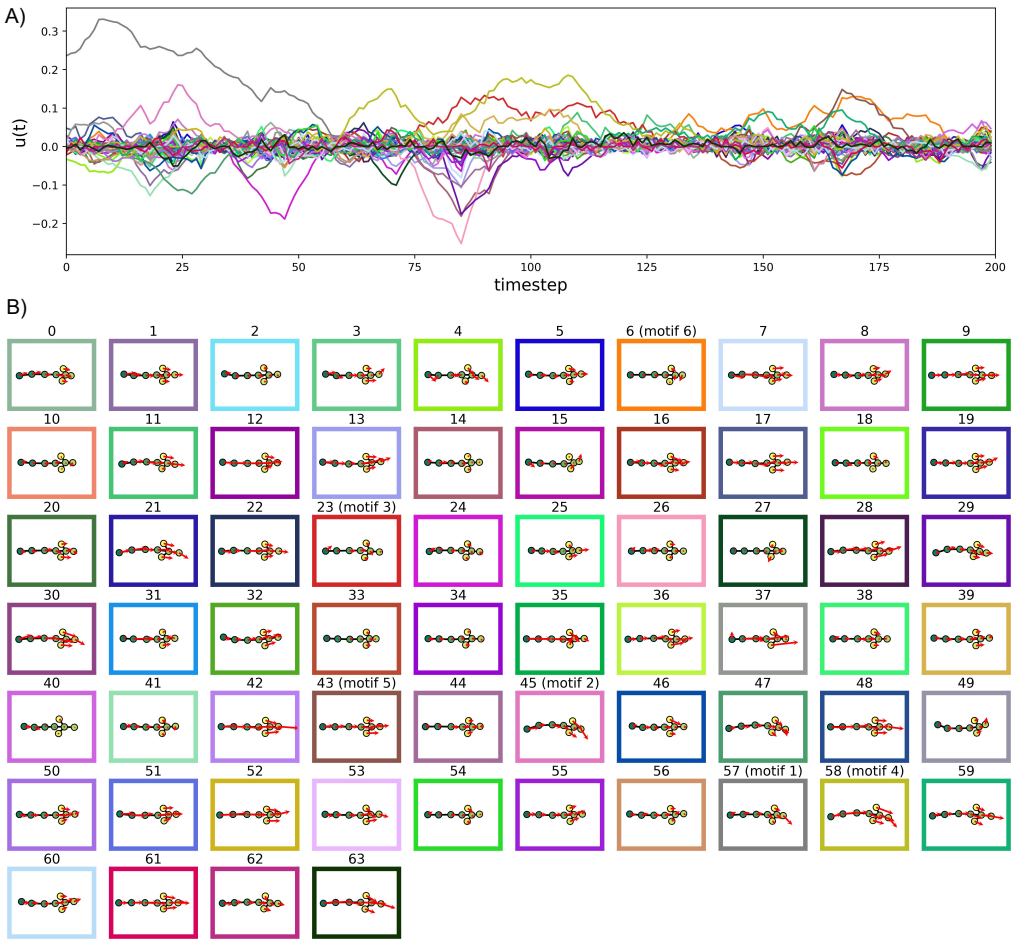

Figure 8: **A.** Policy weight $u(t)$ for all 64 motifs. **B.** The motion field for all 64 motifs learned from the video, computed by averaging the states and actions that most strongly activate each motif.

Table 1: Data subsets constructed from Diego's drawing trajectories. Calibration and test strokes are excluded from MCD training.

Subset	Contents	Purpose
Isolated primitive basis	413 isolated strokes from 19 predefined single-shape types	MCD training and K-means template construction
Practised characters	9,981 strokes from 1,071 character identities	MCD training
Novel characters	2,040 strokes from 665 character identities	Held-out evaluation
Primitive controls	3,577 strokes from separate isolated and grid trials	Similarity-threshold calibration

K Macaque drawing dataset

Task and motivation. We evaluate MCD on the macaque drawing dataset of Tian et al. [35], in which two macaques, Diego and Pancho, copied geometric figures on a touchscreen. The original study investigated whether familiar drawing strokes, or *primitives*, were reused when the animals produced novel characters.

We ask two related questions: whether the learned MCD representation separates familiar primitive strokes into recognizable groups, and whether these groups can be identified within unseen characters. We address the first question by clustering isolated training strokes with K-means, and the second by matching novel-character strokes to the resulting templates. Our analysis uses Diego's behavioral trajectories. No neural recordings enter MCD training or evaluation.

Training, calibration, and test data. We construct four non-overlapping subsets from Diego's released trajectories (Table 1).

The first two subsets are used to train MCD. After training, the isolated primitive basis is used to construct templates by K-means clustering. The primitive controls contain strokes from separate single-primitive and grid-arranged primitive trials. They are reserved for calibrating match thresholds, not for fitting MCD or the templates. The novel-character strokes are used only for evaluation. Thus, the representation, clusters, and thresholds are all established without using the novel-character test set.

Preprocessing and task definition. We downsample each stroke to 25 Hz by retaining every twentieth sample of the original 500 Hz trajectory. We express its two-dimensional coordinates relative to its starting position and apply a coordinate scale estimated from training trajectories only.

For transition t of stroke r, the state $s_{r,t}$ is its stroke-local position and the action is the displacement to the next sampled position, $a_{r,t} = s_{r,t+1} - s_{r,t}$. Stroke boundaries are preserved, and transitions never connect strokes.

MCD uses task-conditioned policy coefficients. For practised-character trials, task identity is the character identity, shared by all strokes of a given character. The isolated-stroke trials contribute 19 additional single-shape task identities, yielding $1{,}071 + 19 = 1{,}090$ training tasks. Task k conditions the policy coefficients $u^{(k)}$ in $Q(s, a \mid k) = \phi(s, a)^{\mathsf{T}} u^{(k)}$, while $\phi(s, a)$ is shared across tasks. The primitive label from the dataset is removed from training.

MCD training. We train a shared 64-dimensional motif representation $\phi(s, a) \in \mathbb{R}^{64}$ with task-conditioned policy learning. For each update, we sample a training task uniformly, then a stroke within the task, and finally a transition within the stroke. This reduces domination by long strokes or heavily represented tasks. We use the checkpoint after 1,000,000 training steps, with batch size 128, feature learning rate 10^{-4}, policy learning rate 3×10^{-4}, and discount factor 0.9. The feature network has no hidden layer. We freeze the learned representation before constructing templates or evaluating novel strokes.

Stroke embeddings and unsupervised templates. We represent each stroke r by averaging its transition-level motif features over its T_r transitions:

$$z_r = \frac{1}{T_r} \sum_{t=1}^{T_r} \phi(s_{r,t}, a_{r,t}). \tag{19}$$

We apply K-means to the 413 isolated training stroke embeddings with $K = 19$, matching the size of Diego's predefined primitive reference library. For each discovered cluster c, the reference template is the mean of its member embeddings:

$$\mu_c = \frac{1}{|\mathcal{B}_c|} \sum_{r \in \mathcal{B}_c} z_r, \tag{20}$$

where $\mathcal{B}_c$ contains the isolated strokes assigned to cluster c.

Clustering uses the mean-pooled embeddings without additional normalization or whitening. We run ten predetermined clustering seeds, with 50 K-means initializations per seed.

Agreement with predefined primitive types. After clustering, to assess whether the discovered clusters correspond to recognizable primitives, we compare them with the predefined types of the isolated strokes. We use Hungarian matching to obtain a one-to-one alignment between cluster identities and primitive types.

Averaged over the ten clustering seeds, MCD achieves 89.8% Hungarian-aligned agreement, an adjusted Rand index (ARI) of 0.840, and normalized mutual information (NMI) of 0.917.

These scores are evaluated on the same 413 isolated strokes used to fit K-means. They measure agreement of the discovered partition with the reference labels. ARI is adjusted for chance and has an expected value of zero under random independent partitions.

Measuring stroke similarity. We compare each stroke embedding with all 19 templates using cosine-based similarity:

$$S(z, \mu) = \left(2 - \frac{z^{\mathsf{T}}\mu}{\|z\|_2 \|\mu\|_2}\right)^{-1}. \tag{21}$$

Equivalently, $S(z, \mu) = 1/(1 + d_{\cos}(z, \mu))$, where $d_{\cos}$ is cosine distance. Higher similarity indicates a closer match.

Each stroke is assigned to its nearest template:

$$\hat{c}_r = \underset{c \in \{1,\ldots,19\}}{\arg\max}\ S(z_r, \mu_c). \tag{22}$$

Every stroke has a nearest template, even when that template is a poor match. We therefore additionally require its similarity to pass a template-specific threshold.

Calibrating high-quality match thresholds. Following the original study's nearest-template matching logic, we estimate a separate similarity threshold τ_c for each template. The thresholds use only the 3,577 independent primitive-control strokes, not MCD training strokes or novel-test strokes.

We first embed each control stroke and assign it to its nearest template. For template c, we collect the similarities of the controls assigned to it:

$$\mathcal{S}_c = \{S(z_r, \mu_c) : r \in \mathcal{C},\ \hat{c}_r = c\}, \tag{23}$$

where $\mathcal{C}$ is the calibration set.

We remove similarities below $Q_{1,c} - 3.5\,\mathrm{IQR}_c$, where $Q_{1,c}$ is the first quartile and IQR_c is the interquartile range of $\mathcal{S}_c$. The cutoff is then the 2.5th percentile of the remaining similarities:

$$\tau_c = \mathrm{Quantile}_{0.025}\left(\mathcal{S}_c^{\mathrm{filtered}}\right). \tag{24}$$

This procedure estimates the similarity range observed for independent primitive controls, while preventing extreme low-similarity outliers from making the cutoff overly permissive. All thresholds are fixed before test evaluation.

High-quality match classification. A novel-character stroke is accepted as a high-quality (HQ) match if its similarity to its nearest template reaches that template's cutoff:

$$h_r = \mathbf{1}\left[S(z_r, \mu_{\hat{c}_r}) \geq \tau_{\hat{c}_r}\right]. \tag{25}$$

Thus, HQ indicates that a stroke passes the calibrated similarity criterion for its assigned template.

Comparison methods. We compare trained MCD with two references.

Untrained MCD uses the same encoder architecture with frozen random parameters. We apply the same K-means template construction, cosine-based matching, and independent threshold calibration as for trained MCD. Both variants use ten clustering seeds.

Label-based DTW uses the original study's released primitive templates and velocity-based dynamic time warping (DTW) procedure. Unlike MCD's K-means templates, this reference library uses predefined primitive labels.

For every method, we independently calibrate its thresholds using the same 3,577 controls and evaluate the same novel-test strokes. This matches the evaluation dataset and the nearest-template acceptance rule, while allowing each method to use its own templates, similarity function, and calibration cutoffs. The label-based DTW baseline uses predefined labels, and is compared with MCD which discovers primitive in an unsupervised way.

Full novel-character test set. The full test set contains 2,040 strokes from 665 novel character identities drawn by Diego. Its HQ rate is the fraction of test strokes accepted:

$$\mathrm{HQ}_{\mathrm{full}} = \frac{1}{|\mathcal{T}|} \sum_{r \in \mathcal{T}} h_r, \tag{26}$$

where $\mathcal{T}$ contains all novel-test strokes.

Shared-character subset. For the main-text figure, we additionally restrict evaluation to character identities drawn by both Diego and Pancho.

Using the released experimental metadata, we first identify groups defined by animal, recording date, and task set that contain both novel and practised character conditions. Among these eligible groups, we find the exact character identities represented for both animals. We then retain Diego's held-out novel-test strokes with those identities. This gives 394 strokes from 135 character units.

Pancho's data are used only to identify shared character identities. His trajectories are not used to train or evaluate the Diego MCD model.

For this subset, we calculate the HQ fraction within each character unit, using the released character-group identifier, and average these fractions with equal weight:

$$\mathrm{HQ}_{\mathrm{shared}} = \frac{1}{|\mathcal{G}|} \sum_{g \in \mathcal{G}} \left(\frac{1}{|\mathcal{T}_g|} \sum_{r \in \mathcal{T}_g} h_r \right), \tag{27}$$

where $\mathcal{G}$ contains the selected character units and $\mathcal{T}_g$ contains the test strokes in unit g.

This gives equal weight to each character unit rather than to each stroke. The full and shared results therefore differ in both the selected drawings and the averaging rule. Templates and calibrated thresholds remain unchanged between subsets.

Aggregation and uncertainty. For MCD and untrained MCD, point estimates are averaged over the ten clustering seeds. For the full test set, we first average each stroke's HQ indicator across seeds and then calculate the overall stroke-weighted rate. For the shared subset, we average the within-character HQ fractions across seeds and then across character units.

We obtain 95% bootstrap intervals by resampling whole character identities for the full test set and whole character units for the shared subset, with replacement. All strokes within a sampled character or unit remain together. The full-set bootstrap recomputes the stroke-weighted estimator, whereas the shared-subset bootstrap recomputes the equally weighted mean of character-unit HQ fractions.

The bootstrap reflects sampling variation across evaluation characters, conditional on the trained representation and the templates and calibration thresholds obtained for each clustering seed. It does

Table 2: Primitive-matching HQ rates on Diego's novel drawings, with 95% bootstrap intervals. All methods use the same evaluation strokes within each subset. MCD and untrained MCD use K-means templates without primitive-type labels. Their results are averaged over ten clustering seeds.

Method	Full test set	Shared subset
Label-based DTW	85.0% (83.5–86.6%)	90.3% (87.0–93.2%)
MCD	80.2% (78.5–81.8%)	86.9% (83.6–90.1%)
Untrained MCD	65.1% (63.1–67.1%)	70.9% (66.2–75.5%)

not quantify uncertainty from independently retraining MCD. The plotted intervals must therefore not be interpreted as variation across MCD training runs.

Results and interpretation. Table 2 reports the HQ rates and 95% bootstrap intervals.

On the shared-character subset, MCD achieves an HQ rate of 86.9%, compared with 90.3% for label-based DTW and 70.9% for untrained MCD. The same ordering holds on the full test set, with rates of 85.0%, 80.2%, and 65.1% for DTW, MCD, and untrained MCD, respectively.

The two evaluations provide complementary evidence: K-means groups isolated-stroke embeddings into clusters that largely agree with the original primitive types, and the same clusters cover a high fraction of strokes within novel character compositions. The trained representation also exceeds the random-encoder baseline under the same clustering and calibration procedure. These findings support the presence and transfer of reusable stroke structure in MCD's learned representation.

L External validity

MCD's result of the navigation labyrinth dataset is based on the task segmentation of [31]. There remains two questions: (1) how do other models perform on the same segmentation; and (2) whether the ground truth from [31] is valid. To further validate our results, we evaluated two additional inverse reinforcement learning baselines: (1) IQ-Learn (IQL, [41]), where we use the same segmentation from [31] to infer a reward map for each task; and (2) Dynamic-IRL (DIRL, [33]), where without relying on this segmentation, we infer a time-varying mixture of reward functions directly from their curated data. We use a tabular version of IQL here just like MCD. For comparison, we constructed two auxiliary reward maps: a perfect reward map with artificially placed rewards, and a random baseline map whose entries are sampled from a standard normal distribution. In the perfect map, the water-seeking task reward function has a peak of 1 at the water port. The home-seeking task reward function has a peak of 1 at the home location. And the exploration task reward function has a uniform low reward of 0.3 at every location.

Fig. 9A shows all inferred maps, and Fig. 9B reports the Pearson correlation coefficients across all maps. There are three important conclusions. (1) The strong alignment between the MCD-recovered maps, the ground-truth maps from [31] (GT), and the perfect manual map demonstrates that MCD successfully recovers the underlying motivational structure. (2) MCD achieves consistently higher correlation with both GT and the perfect map than IQL under the same task segmentation, suggesting that MCD captures the animal's latent motivation more accurately than the IQL baseline. (3) The high correlation between GT, the perfect map, and DIRL provides external validation for the quality of the GT annotations themselves.

Note that although DIRL achieves higher correlations with the ground-truth maps, it is not suitable for our setting. DIRL requires stereotyped, repeated sequences of *identical* length and therefore depends on carefully curated datasets in which such fixed-length segments can be extracted (not the same dataset as we use, but the same environment). In contrast, our datasets—especially the continuous animal free-moving dataset—exhibit substantial heterogeneity and do not contain repeated trajectories of uniform duration. We are also concerned that such curation would introduce bias and distort the natural statistics of the behavior. Furthermore, DIRL relies on *tabular* value iteration to backpropagate gradients from the policy to the reward, which is fundamentally incompatible with the continuous state and action spaces present in the animal free-moving behavior data.

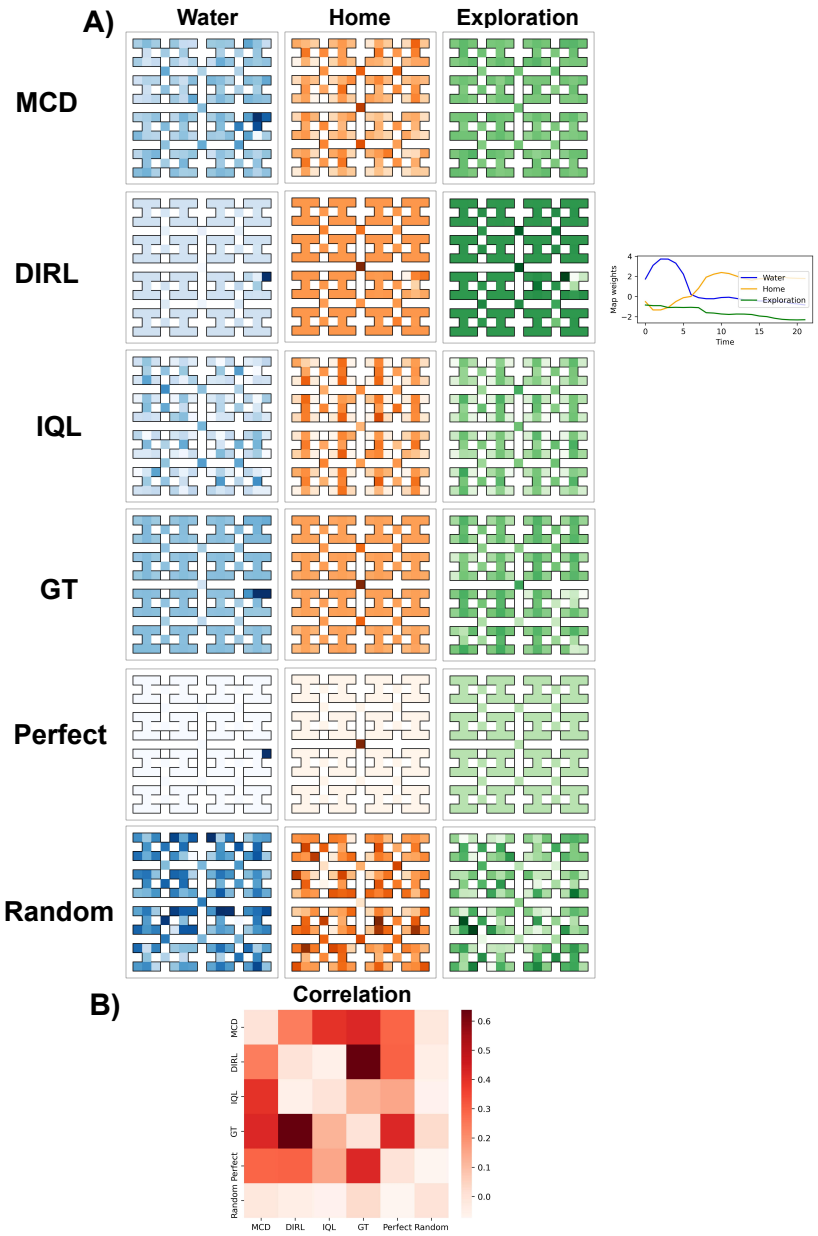

Figure 9: **A.** Reward maps inferred by different models. The maps from DIRL are ranked based on the time-varying weights (right at the corresponding row). "Water" weight increases and then decreases; "home" weight remains low and then increases; and "exploration" weight remains low. **B.** Correlations across the reward maps.

M Lunar Lander dataset

We test MCD on a synthetic continuous Lunar Lander environment. In this environment, the agent controls a rocket descending toward the lunar surface using left, right, and main engines. Each action fires one of the engines, consumes fuel and affects the lander's pose. The final stable landing requires the agent to balance energy efficiency and stability.

State variables. At each time step, the environment state is represented by an 8-dimensional vector

$$(x, y, v_x, v_y, \theta, \dot{\theta}, \text{leg}_L, \text{leg}_R),$$

where (x, y) denote position, (v_x, v_y) linear velocities, θ the lander's angle, $\dot{\theta}$ its angular velocity, and $\text{leg}_L, \text{leg}_R \in \{0, 1\}$ indicate ground contact of the left and right landing legs. The *contact* is defined as whether either leg is in contact with the ground.

The agent selects a discrete action $a_t \in \{0, 1, 2, 3\}$ at each time step, corresponding to:

- 0: do nothing,
- 1: fire the left orientation engine,
- 2: fire the main engine,
- 3: fire the right orientation engine.

Rewards increase when the lander achieves a smooth and upright touchdown with minimal oscillation, and decrease when tilting, excessive acceleration, or rolling happens.

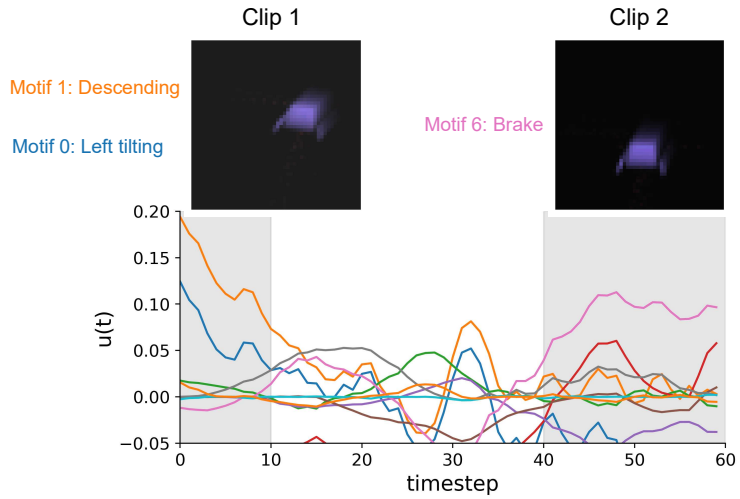

Figure 10: Motif weights for a clip from the lunar lander dataset.

We use the continuous formulation in Sec. 4.2 to learn motif representations $\phi(s, a)$ and time-varying weights $u(t)$ (Sec. 4.3). This dataset contain trajectories generated by a trained agent ($N = 62544$). To encourage temporal smoothness, we place a Gaussian random-walk prior on $u(t)$: $u(t) \sim \mathcal{N}(u(t-1), \sigma^2 I)$. We optimize $u(t)$ after learning motifs. The motif dimension is set to $d = 64$.

We select a representative clip to illustrate interpretability. The first segment exhibits rapid descent with a left tilt, while the second shows braking with smoother descent. This transition is captured by a mixture of two motifs in the first segment and the dominance of Motif 6 in the second (Fig. 10). We attach the video in the supplementary material.

To interpret the motifs, we collect state–action pairs that maximally activate each motif, carry out PCA, and visualize the PC motifs' distributions in Fig. 11.

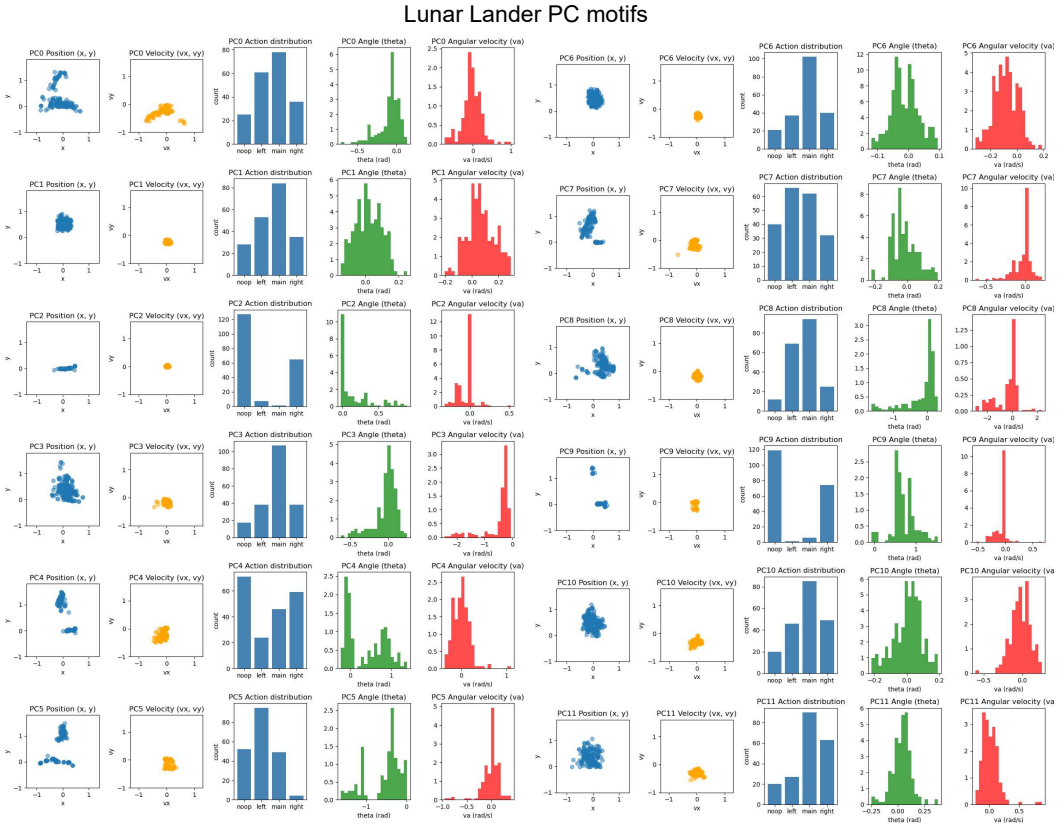

Figure 11: For each PC motif, we collect the data entries that most strongly activate it, and show the distribution of positions, velocities, angle, and angle velocities.

N Evaluation details on a human-annotated dataset.

We verify our method on a supervised behavior label dataset. Due to the substantial cost and labor demands of manual annotation, large-scale, high-quality animal behavior datasets with human labels remain limited. Commercial annotation tools (e.g., HomeCageScan) are also expensive, further constraining availability. To support quantitative evaluation, we therefore included a human-annotated subset from the freely moving animal behavior dataset we mentioned last section. Specifically, we selected 200 short video clips (10 recording sessions; in each session we randomly sampled 20 clips, with 200 frames each). Each clip was manually annotated with behavior segments for each frame, resulting in 200 per-frame labels across six categories: `Walking`, `Sniffing/Grooming`, `Turning left`, `Turning right`, `Rearing`, and `Resting`. Manual annotation required approximately 40 hours of expert effort. The dataset is split into a training set and a test set with a ratio $8:2$.

As before, we evaluated four models on this dataset: MCD, Keypoint-MoSeq, SemiSeg, and OPAL. Because the latent variables or syllables inferred by these models do not necessarily align with human-defined labels, we trained an additional decoder to map model-specific latent representations to the ground-truth categories. For each model, the decoder input was the motif weights $u(t)$ (MCD), the inferred one-hot syllable (Keypoint-MoSeq), or the latent embedding z (SemiSeg and OPAL). The decoder was implemented as a two-layer neural network with ReLU activation, optimized using the Adam optimizer and cross-entropy loss (learning rate $= 0.001$, hidden size$= 100$), following standard settings in the scientific computation package `scikit-learn`. The resulting classification accuracies on the held-out test set (Fig. 4A) demonstrate that MCD substantially outperforms all baselines, achieving near-perfect accuracy. This indicates that the motifs recovered by MCD align better with actual animal behavior dynamics.

O Connections to relevant algorithms in RL

Motor primitives learning. Animal behavior research has focused on identifying motor primitives directly from large-scale recordings. Supervised classification [11, 12], clustering-based analysis [4, 5], and HMM-based methods [1, 2, 3, 13] label behavior into discrete syllables and analyze neural correlates of transitions. While these approaches reveal stereotyped motifs such as walking or grooming, they typically treat primitives as rigid and exclusive, overlooking the continuous, compositional structure of behavior and its adaptation across contexts.

In robotics, movement primitives have long been studied [42, 43, 44]. ProMP [42] extracts primitives from demonstrations but requires labeled skills/motifs, while ProbS [44] jointly infers behavioral segmentation and a primitive library via Expectation-Minimization (EM), enabling unsupervised skill discovery. These approaches resemble dynamics-based behavior segmentation in animal behavior research.

All methods mentioned here lack an RL perspective that links primitives to policies and reward-driven behavior. Our framework instead leverages offline RL-based imitation learning to uncover motif-based policies that capture both modular primitives and their sequential composition, and provides a principled way to explain why behaviors occur, not just how they unfold, and how reusable motifs/skills contribute to them in a generative decision-making process.

Offline Imitation learning. Offline imitation learning presents the problem of learning a policy from fixed demonstrations when access to environments is impossible. Simple behavior cloning can be performed offline, but fails to generalize well in some cases because it does not consider dynamics or environment structure limits. ValueDICE [45] considers the dynamics of the training data and learns a policy that minimizes the KL-divergence between the state-action occupancies generated by the policy and of the original dataset. But the adversarial optimization of policy and Q-functions introduces instability in training. IQL [41] avoids adversarial training by directly parameterizing the policy in terms of the Q-function, $\pi(a|s) = \exp(Q(s,a))/Z$. However, all of these algorithms learn policies as unstructured functions. As the animal behavior structure is highly modular and stereotyped, it is more appropriate to employ a hierarchical motif-based policy to model the data.

Motif/Skill discovery. Unsupervised skill discovery (DIAYN [16], BeCL [46], DADS [17], InfoGAIL [47], Directed-InfoGAIL [48], SkillBlender [49], ASE [50]) to find a high-level abstraction of actions has been an effective strategy for online RL and imitation learning. However, they are limited to online settings when the skills can only be refined through interacting with the environment, which restricts their application to large-scale offline datasets. Recent offline skill discovery methods include OPAL [18], SPiRL [19], and SkiLD [51] which use an autoencoder to encode trajectories into latent skills z; and PARROT [52] which uses a flow-based model to learn a behavior prior. In these models, the latent z is later used to generate the policy $\pi(a|s,z)$ in a non-linear way. Our paper, instead, employs a generalized-linear structure of the policy $\pi(a|s,z) \propto \exp(\phi(s,a)^\top u(z=t))$ which provides better interpretability than the policy network. This interpretability is essential for addressing downstream neuroscience questions.

Linear structure of environment/policy. Another line of work (HILP [53], FB [54], USFA [55], RaMP [56]) based on successor features (SF) also uses a generalized linear structure to model the motif-based policy as $\pi(a|s,z) \propto \exp(F(s,a,z)^\top z)$ where $F(s,a,z)$ is SF under a certain motif z. The main concern is that they cannot separate motif representation z from state-action representation in SF, while our work can (Eq. 4). Therefore, their motifs depend on the specific task or timepoint, while our motifs/skills are general representations shared across tasks and timepoints. The idea that task-agnostic motifs are combined adaptively to form new policies aligns better with the need of interpretability and scientific discovery: we would like to look for neural signals responsible for relatively fixed time-agnostic behavior patterns, to help us better understand the animal behavior.

TRAIL [57] adopts a linear decomposable environment as MCD when learning the latent skills. However, the latent-conditioned policy $\pi(a|s,z)$ is not linear, but parametrized by a neural network. SkillBlender [49] adopts a linear decomposable policy, and uses linear combinations of lower-level controller outputs as the final actions $a_t = \sum_i k_i a_t^i$, while we use the linear combinations of different motifs to generate the final state-value functions $Q_t(s,a) = \sum_i \phi_i(s,a) u_t^i$ rather than direct actions. We believe a naive mixture of the low-level controller is less biologically realistic in modeling animal behavior than mixtures of state-value functions. The latter can find supporting evidence in neuroscientific literature [58]. Besides, their transition kernel $P(s'|s,a)$ is not linear.

Compared to them, our work assumes generalized linear structures for both the policy and the environment, providing better interpretability. This shared motif is more fundamental in revealing the basic structure of the animal's intentions.

Our contributions. Most existing skill-learning methods do not apply to the behavior understanding scenario considered here. We study offline data without supervision or task annotations—no explicit goals, labels, or trajectory segmentation—only long, unstructured behavior sequences. The challenge is threefold: (1) to discover the basic motor skills, (2) to determine how these skills compose spatially and temporally within long trajectories in an offline setting, and (3) to ensure interpretability for scientific discovery. These are nontrivial problems beyond the reach of existing skill- or motor-primitive approaches. MCD addresses these challenges within an offline RL-based imitation learning framework, yielding interpretable motif representations and their continuous compositions.

P Identifiability discussion

In this section we clarify the notion of identifiability and how it applies to our motif-based continuous dynamics (MCD) model. A parameterization is said to be *identifiable* if different parameter values lead to different model predictions. Conversely, if multiple distinct parameterizations yield identical likelihoods or value functions, the model is only identifiable up to an equivalence class (e.g., scaling, rotation, or permutation).

Our model is identifiable at the subspace level. In MCD, both the transition operator and the action-value function admit the spectral form

$$P(s'|s,a) = \phi(s,a)^\top \mu(s'), \qquad Q(s,a) = \phi(s,a)^\top u.$$

Although individual components of ϕ, μ, and u may be rescaled by an invertible linear transformation without altering P or Q, the *subspace spanned by the motif functions* is uniquely determined [9]. This is analogous to PCA, where the principal subspace is identifiable even if the basis vectors within that subspace are not uniquely determined. In practice, they converge well.

Moreover, in the continuous version, the identifiability of the new representations ψ, ν up to orthogonal transformations has been guaranteed by the normalization operation before the output (Appendix F), which also ensures the trainability of the contrastive learning algorithm.

Comparison to HMMs and latent variable models. Classical behavioral segmentation methods (e.g., Keypoint-Moseq as an HMM variant, SemiSeg as an autoencoder variant) are generally not identifiable without strong additional assumptions. Different transition matrices, emission matrices, or latent dynamics parameters can produce indistinguishable observation distributions. These models typically suffer from label-swapping and more severe transform equivalences in the latent space, meaning that even the latent subspace is not uniquely determined. In contrast, our method fixes the ambient representation through the observed state–action pairs and learns a function class whose span is uniquely tied to the transition operator and value function. Thus, while our motif basis is identifiable only up to linear transforms (as is standard for spectral decompositions), the underlying subspace and its behavioral interpretation are stable, reproducible and identifiable.

Summary. MCD inherits the identifiability properties of spectral RL methods: the motif basis is identifiable up to an invertible transformation, and the motif *subspace*—the structure that determines P, Q, and π—is uniquely recoverable. This stands in contrast to latent-variable models such as HMMs or SLDS, which in general do not admit identifiable latent subspaces without strong, often unrealistic, constraints.

Q Long-term dependency discussion

Compared with prior approaches, our framework implicitly captures the long-term temporal structure of behavior through the Bellman equation in the reinforcement learning (RL) formulation. The component responsible for encoding long-term and multi-scale dynamics is the motif weight $u(t)$. From the linear decomposition of the state–action value function (Eq. 3),

$$u(t) = w + \gamma \int V(s')\,\mu(s')\,q(s')\,ds',$$

it follows that $u(t)$ integrates two sources of information simultaneously: (i) long-horizon structure through the value function $V(s')$, which summarizes discounted future trajectories with an infinite horizon, and (ii) short-horizon structure through the immediate reward parameter w. Therefore, this formulation naturally enables modeling of multi-scale temporal dependencies. Empirically, we observe this structure in Fig.4B: some motifs exhibit stable, slowly varying activation profiles (e.g., motif 1, gray curve), whereas others fluctuate rapidly and capture short-term transitions (e.g., motif 2, pink curve). Intuitively, consider a mouse that intermittently sniffs throughout the day but engages in fast running only during a brief morning period. The coefficient $u_i(t)$ for sniffing would rise gradually across long time windows, while the coefficient $u_j(t)$ for running would increase only during short, specific intervals.

R Continuous composition versus smoothing of discrete motifs

Question and experimental setup. A smoothly changing representation does not necessarily imply continuous composition. It could instead represent one discrete state at a time, with smoothing creating mixtures only when the state changes. We test this explanation using two complementary analyses: where mixing occurs in time, and how much behavioral information remains after replacing MCD with discrete states.

We use the human-annotated free-moving mouse dataset, containing 200 clips of 200 frames each at 30 fps and six behavior categories. We reuse the trained MCD model and its inferred 64-dimensional representation $u(t)$. MCD is not retrained for this analysis.

Mixing within stable bouts and at behavior boundaries. To measure the mixing of $u(t)$, we normalize their absolute magnitudes into relative component weights:

$$p_j(t) = \frac{|u_j(t)| + \epsilon}{\sum_{k=1}^{D} \big(|u_k(t)| + \epsilon\big)}, \qquad D = 64, \quad \epsilon = 10^{-12}. \tag{28}$$

and quantify mixing using normalized Shannon entropy and top-1 mass:

$$H(t) = -\frac{\sum_{j=1}^{D} p_j(t) \log p_j(t)}{\log D}, \qquad M_1(t) = \max_j p_j(t). \tag{29}$$

Entropy is near zero when one component dominates and increases as weight is spread across components. Top-1 mass measures the share carried by the largest component.

We call a frame a boundary frame if it lies within ± 5 frames (approximately ± 167 ms) of a change in the human behavior label. All remaining frames are stable-bout interiors. A stable bout means that the annotated behavior stays the same.

For comparison, we encode the six human behavior labels as one-hot states. We smooth this sequence within each clip using a Gaussian kernel with $\sigma = 3$ frames (100 ms). This creates a smoothed discrete labels baseline whose mixing is concentrated around behavior changes. Its entropy is normalized by $\log 6$, so that both representations have entropy on a zero-to-one scale.

As shown in Table 3, the smoothed discrete baseline is almost entirely dominated by one component inside stable bouts, with mixing mainly near boundaries. In contrast, MCD has nearly the same entropy in both regions. Its largest component carries approximately 11% of the absolute weight near boundaries and in stable interiors. This pattern remains qualitatively unchanged for boundary half-windows of 1, 3, 5, 10, and 15 frames. Thus, MCD's mixing is not concentrated at the human-annotated behavior transitions.

Table 3: Frame-weighted mixing statistics near behavior changes and within stable bouts. Boundary frames use a ± 5-frame window.

Representation	Boundary entropy	Interior entropy	Interior top-1 mass
Continuous MCD $u(t)$	0.7915	0.7877	0.1126
Smoothed discrete labels	0.2640	0.0062	0.9978

Does smoothing discrete states preserve behavioral information? High entropy alone does not establish that mixing is useful. We therefore ask whether discrete states with smoothed transitions can preserve the human-label information available in continuous $u(t)$.

For each decoding split, we fit a 64-cluster K-means quantizer to the training-frame representations, without using human labels. Each frame is then represented by a one-hot code indicating its nearest cluster. We compare these hard codes with a version smoothed within each clip using the same $\sigma = 3$-frame Gaussian kernel. The number of discrete states matches the 64 components of MCD. Unlike the label-based diagnostic above, these discrete codes are obtained from MCD rather than from the decoding targets.

We use the same decoding protocol for all three representations: an MLP with one hidden layer of 100 units, matched random $80/20$ frame-level train/test splits, and ten random seeds. Table 4 reports mean accuracy and standard deviation across seeds. The majority-class baseline is the accuracy obtained by always predicting the most frequent behavior label; it is 0.5566 for these data.

Table 4: Human-label decoding accuracy. Model results are mean ± standard deviation over ten seeds.

Representation	Decoding accuracy
Continuous MCD	0.9615 ± 0.0018
Hard 64-state K-means	0.5761 ± 0.0099
Smoothed 64-state K-means	0.6216 ± 0.0056
Majority-class baseline	0.5566

Smoothing improves the discrete baseline from 0.5761 to 0.6216, but both remain far below continuous MCD (0.9615). Smooth transitions therefore help, but do not recover most of the decoding performance lost through this discretization.

Interpretation. The two analyses provide complementary evidence. Mixing persists within stable annotated behaviors, and the continuous representation retains behavioral information that the tested hard and smoothed discrete codes lose. These results support graded, behaviorally informative variation throughout behavior bouts, rather than mixing explained primarily by smoothed transitions between discrete states.

S Hyperparameter sensitivity

S.1 Motif dimension

To evaluate the sensitivity of our model to the dimension of the hidden state, we change the dimensions of motifs d and verify them in different datasets.

In the discrete case, there exists a minimum dimension that could fully represent the motif space. In fact, the motif discovery (Eq. 2) is equivalent to SVD of the transition matrix $P(s'|s,a)$[9]. Since the rank of the transition matrix is $|\mathcal{S}|$, the ranks of ϕ and μ matrices are both $|\mathcal{S}|$, which is exactly the minimum dimension of $\phi(s,a)$ that could fully represent the motif space. This could be clearly seen in the gridworld dataset (Fig. 12) and labyrinth navigation dataset (Fig. 13).

For the gridworld dataset, $d_{min} = 3 * 3 = 9$. If $d < 9$, the SVD could only provide a low-rank approximation. As a result, the recovered reward map is blurred and incorrect (Fig. 12A-C) and the correlation remains low (Fig. 12G). For $d \geq 9$, the motif matrix is full-rank. So the reward maps are nearly perfect (Fig. 12D-F), and the correlations are near 100% (Fig. 12G). Similar analysis also applies to labyrinth navigation dataset where $d_{min} = 127$(Fig. 13). In this case, variant with fewer motif dimensions ($d = 64$) cannot generate realistic reward maps while variants with more motif dimensions ($d \geq 127$) can faithfully recover the reward peaks in each task.

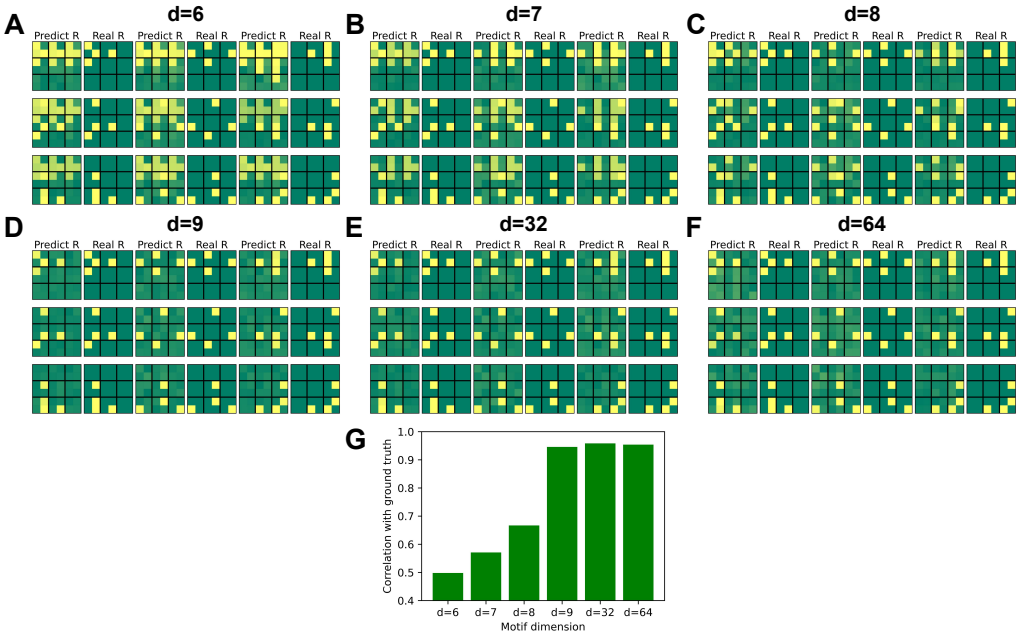

Figure 12: Reward maps generated by models of different motif dimension d in the gridworld dataset.

In the continuous case, since the transition kernel is infinite-dimension, any finite d would only provide a low-rank approximation to the transition kernel. As d increases, the fitting performance would be better. However, since the importance of different motifs is different, by observing the performance under different d, we could select one to be as small as possible while still capturing the entire motif space as comprehensively as possible, and maintain the performance. In this experiment (Fig. 14), we select $d = 64$.

S.2 Noise distribution

The continuous version of MCD is based on contrastive learning. Therefore, a high-quality negative sample distribution is crucial for the motif and policy learning. Here we evaluate different selections of noise distributions in the continuous free-moving animal behavior dataset.

For the motif learning (Eq. 8), in the main text, we directly sample states from the dataset. In other words, the noise distribution is $\rho(s) = \int \tau^e(s,a)da$.

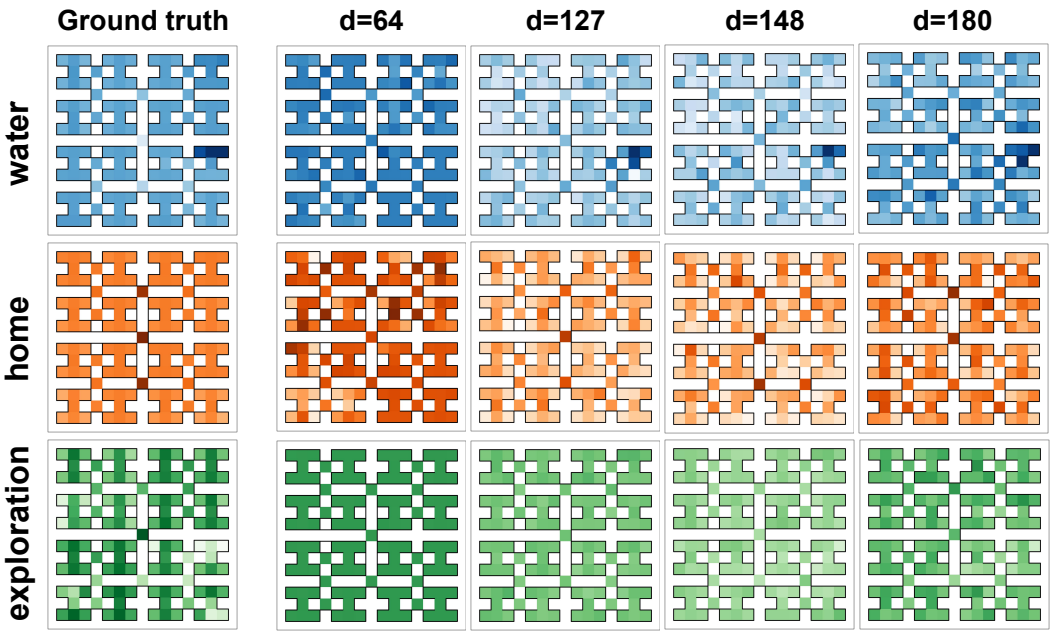

Figure 13: Reward maps generated by models of different motif dimension d in the labyrinth navigation dataset.

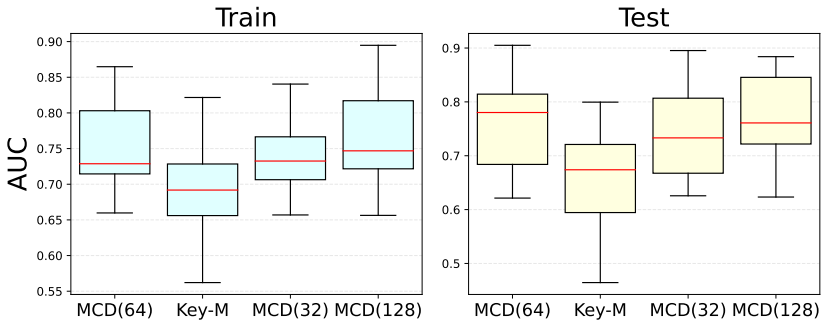

Figure 14: AUC generated by models of different motif dimension d in the animal free-moving dataset.

In "Motif-N" variant, we replace it by a uniform distribution, whose dimension-wise bounds are determined as the maximum and minimum values of the states in the dataset, i.e. $s'_i \sim U(\min(s_i), \max(s_i))$. Because the negative samples are not good enough, the resulting energy-based model can neither capture high-quality motifs nor provide good basis vectors for later policy learning. Therefore the AUC score remains low (Fig 15).

For the policy learning (Eq. 9), in the main text, we also sample negative samples directly from the dataset $(s', a') \sim \tau^e$ and use a' as a negative sample of actions. In other words, the noise distribution is $\zeta(a) = \int \tau^e(s, a)ds$.

In the "Policy-N" variant, we sample negative pairs from uniform distribution $a'_i \sim U(\min(\zeta(a_i)), \max(\zeta(a_i)))$, similar to "Motif-N". In the "AR-Actor" variant, we additionally train an auxilliary autoregressive actor $\pi(a|s)$ to fit $\pi^e(a|s)$ using MLE. And the negative samples are from this actor $\pi(a|s)$. For each dimension, the actor has a network of two hidden layers, with hidden dimension=256 each, to produce the mean and variance of this action dimension, i.e. $a_i \sim \mathcal{N}(\mu(a_i|s, a_{<i}), \sigma^2(a_i|s, a_{<i}))$. As expected, uniform noise distribution is not good enough, and "Policy-N" achieves lower AUC than the original version. And the "AR-Actor" achieves similar performance on the test dataset, proving the robustness of MCD to noise distribution if it is good enough.

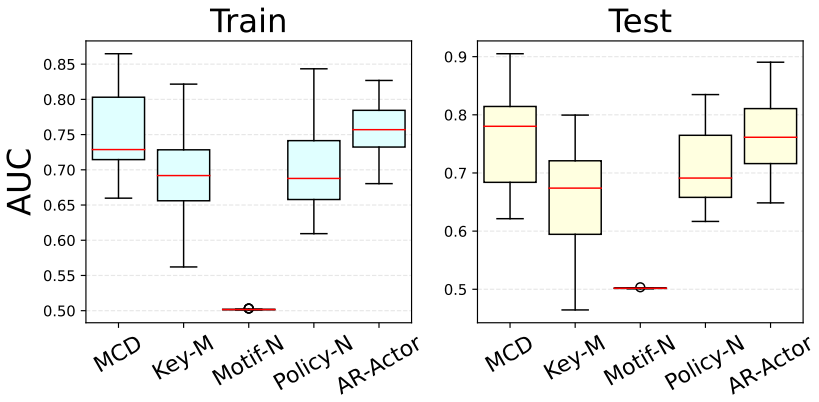

Figure 15: AUC generated by models trained under different noise distribution in the animal free-moving dataset.

We also tested the influences of the number of negative samples (Fig. 16). MCD is consistently better than Key-Moseq and maintains a stable performance. It turns out that the model is robust against the choice of number of negative samples.

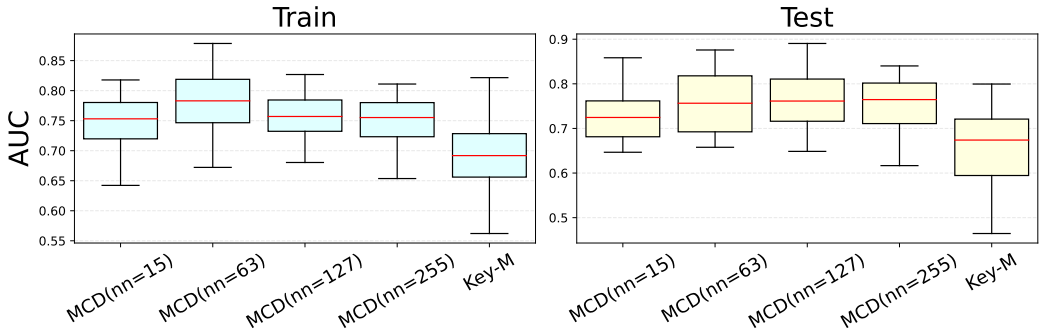

Figure 16: AUC generated by models trained using different numbers of negative samples for motif contrastive learning in the animal free-moving dataset. $nn = k$ means the **n**umber of **n**egative samples per positive sample is k.

S.3 Dataset partition

In the third experiment, we randomly split the whole dataset into training set and test set. Here, we train the model on some mice and then test it on other heldout mice, i.e. split the dataset by mouse identities. This experiment shows whether the motifs learned by the model are general and transferrable across different subjects. The result is shown in Fig. 17 where the complete exclusion of one animal from the training set does not impair the performance. This experiment proves that the MCD can learn a general set of motifs that could be transferred across animals.

S.4 Gaussian random walk prior

In the free-moving animal behavior dataset (Sec. 5.3), we impose a Gaussian random-walk prior on the time-varying motif weights $u(t)$ to encourage temporal smoothness. To assess the sensitivity of our method to this regularization, we varied the strength of the Gaussian prior and re-evaluated model performance (Fig. 18). As expected, very large coefficients introduce slight degradation due to oversmoothing. However, across all tested values, performance remains substantially higher than the baseline model. This demonstrates that MCD is robust to the choice of Gaussian random-walk regularization strength and does not rely on fine-tuning this hyperparameter to achieve strong results.

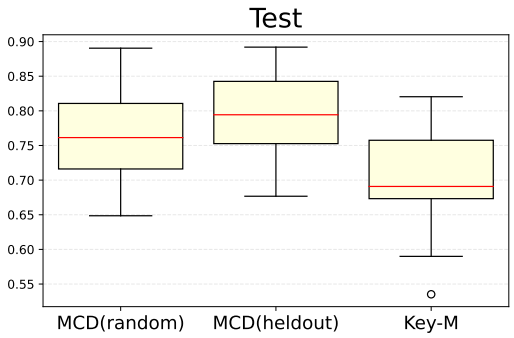

Figure 17: AUC generated by models trained using different train-test split methods. For MCD(random), the dataset is randomly split. For MCD(heldout), the behavior of one mouse is specially held out as the test set while the rest is taken as the training set.

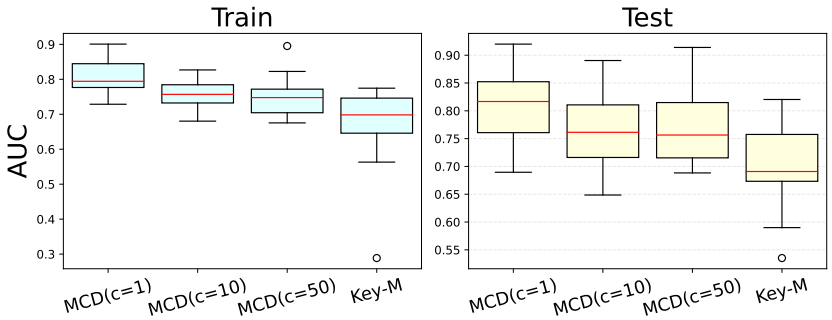

Figure 18: AUC generated by models tested using different Gaussian random walk prior coefficients in the animal free-moving dataset.

S.5 Temporal resolution

We evaluate the sensitivity of MCD to temporal resolution by downsampling the original 30 fps videos to 15, 10, and 7.5 fps, with actions recomputed between new frames. We then use the same MCD checkpoint trained at 30 fps to test their performance (Fig. 19). As in the main text, we report two metrics. **Human-label decoding accuracy** tests whether the fitted MCD representation preserves behaviorally meaningful information. **Test action-likelihood AUC** measures whether MCD assigns higher likelihood to the observed action than to mismatched actions. Chance AUC is 0.5. We also use **Motif space CKA** to measure how different the motif spaces are across downsampling rates.

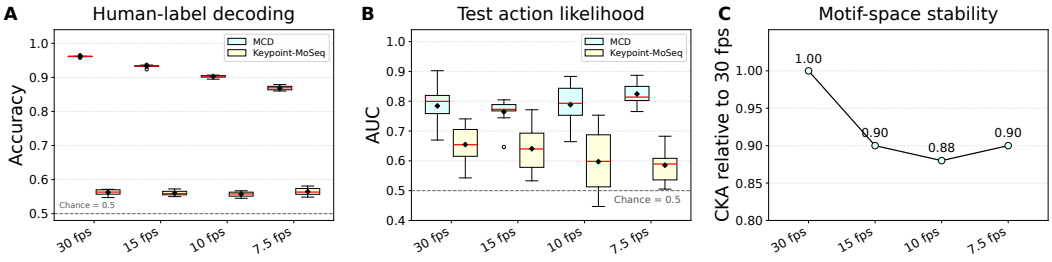

Figure 19: Decoding accuracy (**A**) and AUC (**B**) generated by models tested using different downsampling rates in the animal free-moving dataset. **C.** CKA under different downsampling rates for MCD in the same dataset.

Human-label decoding accuracy decreases gradually as temporal resolution is reduced, while remaining competitive with other baselines. In contrast, test action-likelihood AUC remains well above chance and shows no degradation. The CKA shows only minor decrease, suggesting that the motif space is stable across sampling rates and captures action-relevant information. Therefore, MCD is robust to temporal resolution.

T Generated rollout trajectories

In the animal free-moving dataset, we train a generative energy model to estimate the state–action value function. Here, we assess its reliability using Hamiltonian Monte Carlo (HMC) sampling. For each motif $i \in 1, ..., d$, we construct a one-hot vector u with $u_i = 1$ and $u_j = 0$ for all $j \neq i$, and then sample an action a from the learned energy model. For each motif, we sample 10 actions in total. The resulting motif-specific rollout trajectories (Fig 20) largely align with the empirical averages (Fig. 8B), demonstrating that the model captures the dominant behavioral tendencies. A small number of trajectories deviate from the empirical trends, probably because u is never used as a strict one-hot vector during training but instead appears as a mixture over motifs. We attribute these deviations to unavoidable generative noise and the distributional mismatch between test-time one-hot inputs and the mixed representations observed in real data.

For that, we choose several clips and generate the rollouts (number of rollout steps=5) based on the inferred $u(t)$. The alignment between generated trajectories and real trajectories (Fig. 21) show that MCD could capture the real behavior dynamics.

HMC parameters. We use a step size of 1×10^{-4}, 100 leapfrog steps per action, temperature $= 1$, mass matrix $M = I$, and initial momentum $r \sim \mathcal{N}(0, 1/200)$. The leapfrog integrator is employed to improve sampling accuracy.

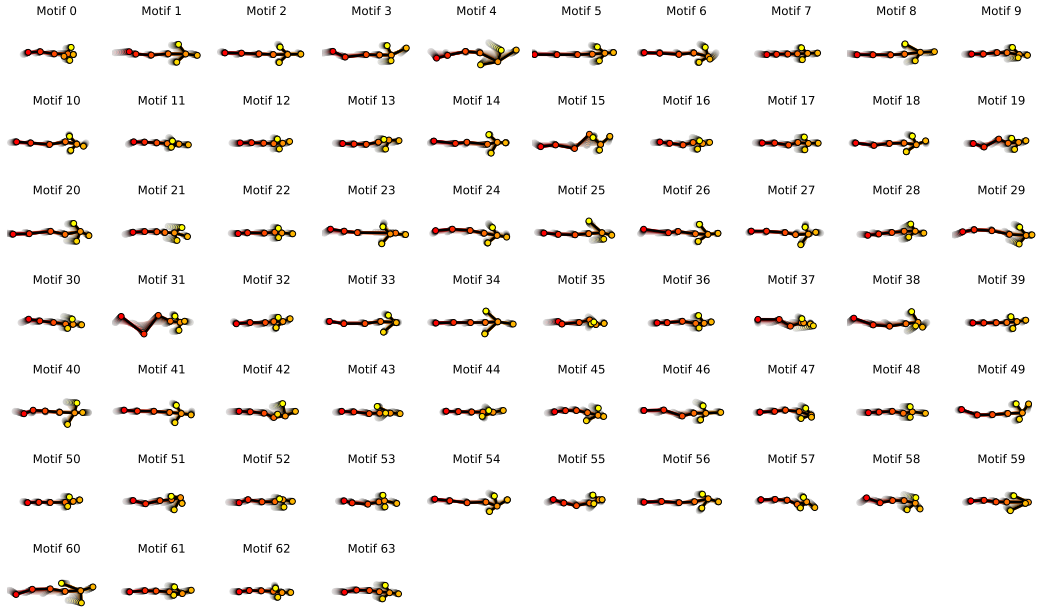

Figure 20: Motif-specific rollout trajectories generated by HMC sampling from the trained energy model. Indexes are consistent with Fig. 8.

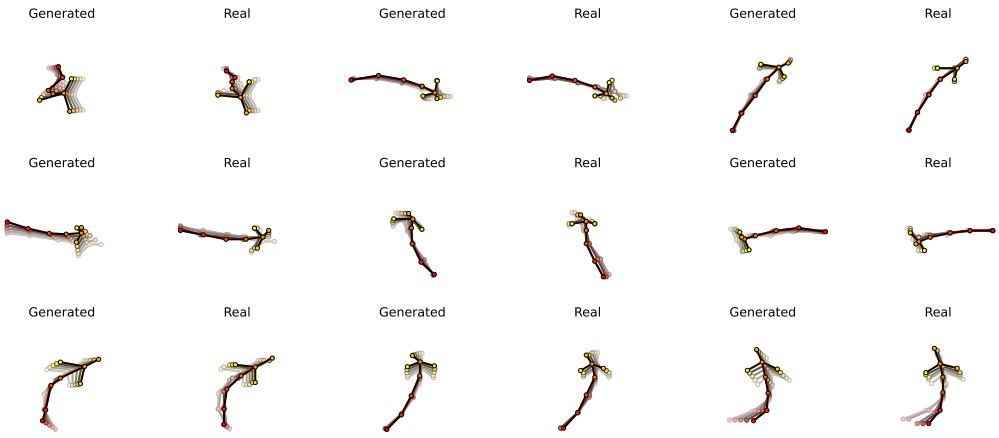

Figure 21: Rollout trajectories based on inferred $u(t)$ generated by HMC sampling from the trained energy model.

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: In Sec. 4, building on results in reinforcement learning and imitation learning, we propose the theoretical framework of our model MCD for behavior understanding. In Sec. 5, we apply our methods to three different datasets: simulated grid world trajectories, trajectories of a real mouse navigating in a labyrinth, the poses of a freely moving mouse, and macaque drawing behavior. We present the interpretation of our results and show that the performance of our results is better than similar models.

   Guidelines:

   - The answer [N/A] means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A [No] or [N/A] answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: The paper explicitly discusses limitations in Sec. 6, including sensitivity to input data quality (e.g., tracking errors) and challenges in validating correspondence between learned motifs and neural mechanisms.

   Guidelines:

   - The answer [N/A] means that the paper has no limitation while the answer [No] means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory assumptions and proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: We define motif sets in Definition 1, and combine it with the reinforcement learning framework to describe the learning procedure of skill sets $\phi(s, a)$ and motif weights $u(t)$. In Sec. 4.2, we solve the difficulty in the continuous dataset using energy-based model (EBM) and prove the connection between EBM and our definition of motif sets in Appendix G. We also derive how to infer reward functions under our framework in Sec. 4.4.

   Guidelines:

   - The answer [N/A] means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental result reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The paper provides sufficient details to reproduce the main experimental results, including theoretical formulations, model architectures, training objectives, dataset construction, evaluation protocols, hyperparameter settings, and compute resources in the main text and appendix. In addition, the supplementary material includes the implementation code used for training and evaluation.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - If the paper includes experiments, a [No] answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.

(c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: We provide our code in the supplemental material as well as the code link.

   Guidelines:

   - The answer [N/A] means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so [No] is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://neurips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental setting/details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer) necessary to understand the results?

   Answer: [Yes]

   Justification: The paper provides sufficient experimental details to understand and reproduce the results, including dataset construction and evaluation protocols in Sec. 5, as well as comprehensive hyperparameter settings, model architectures, and training procedures in Appendix F. Additionally, the code and preprocessing details are included in the supplementary material.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment statistical significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

Answer: [Yes]

Justification: In Fig. 4, we use boxplots to summarize variability in human-annotated label prediction accuracy and AUC across test evaluations. To support the main performance comparison, we additionally perform paired t-tests against each baseline and report the resulting p-values in Sec. 5.3.

Guidelines:

- The answer [N/A] means that the paper does not include experiments.
- The authors should answer [Yes] if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g., negative error rates).
- If error bars are reported in tables or plots, the authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments compute resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [Yes]

   Justification: The paper specifies compute resources in Appendix E, including hardware (CPU, GPU, memory), software framework, and per-experiment training time across all datasets.

   Guidelines:

   - The answer [N/A] means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code of ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: We have reviewed the NeurIPS Code of Ethics and confirm that our work complies with its requirements. Our research uses publicly available or standard behavioral datasets without personal identifiable information, does not involve human subjects or sensitive data collection, and poses minimal risk of harmful application. Potential societal impacts, including both benefits and risks, are discussed in the Impact statement (Appendix A), and no aspect of the work relates to disallowed areas such as surveillance, discrimination, or weaponization.

Guidelines:

- The answer [N/A] means that the authors have not reviewed the NeurIPS Code of Ethics.
- If the authors answer [No], they should explain the special circumstances that require a deviation from the Code of Ethics.
- The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader impacts**

    Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

    Answer: [Yes]

    Justification: The paper discusses broader impacts in the Impact statement (Appendix A), including positive contributions such as advancing understanding of animal behavior and potential applications in neuroscience, as well as negative risks such as possible misuse when extending the framework to model human behavior (e.g., bias amplification). These considerations provide a balanced discussion of both beneficial and harmful societal implications.

    Guidelines:

    - The answer [N/A] means that there is no societal impact of the work performed.
    - If the authors answer [N/A] or [No], they should explain why their work has no societal impact or why the paper does not address societal impact.
    - Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
    - The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate Deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
    - The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
    - If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

    Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pre-trained language models, image generators, or scraped datasets)?

    Answer: [N/A]

    Justification: The paper does not release models or datasets that pose a high risk of misuse (e.g., large pre-trained models, generative systems, or sensitive datasets). Our work focuses on methodological contributions evaluated on controlled and standard behavioral datasets, and therefore does not require specific safeguards for responsible release.

    Guidelines:

    - The answer [N/A] means that the paper poses no such risks.
    - Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring

that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.

- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

    Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

    Answer: [Yes]

    Justification: All datasets, environments, and baseline methods used in this work are properly credited through citations in the main text and references. Their licenses and terms of use, including the Harvard non-commercial academic license for Keypoint-MoSeq and the MIT licenses for the labyrinth dataset and Gymnasium environments, are explicitly documented in Appendix D.

    Guidelines:

    - The answer [N/A] means that the paper does not use existing assets.
    - The authors should cite the original paper that produced the code package or dataset.
    - The authors should state which version of the asset is used and, if possible, include a URL.
    - The name of the license (e.g., CC-BY 4.0) should be included for each asset.
    - For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
    - If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
    - For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
    - If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [Yes]

    Justification: The paper introduces new code implementing the proposed MCD framework, which is provided in the supplementary material and the link. We document training procedures, model architectures, and hyperparameters in the appendix, enabling reproducibility and proper use of the released assets.

    Guidelines:

    - The answer [N/A] means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and research with human subjects**

Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

Answer: [N/A]

Justification: The paper does not involve crowdsourcing or research with human subjects. All experiments are conducted on simulated environments and existing animal behavior datasets.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
- According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional review board (IRB) approvals or equivalent for research with human subjects**

Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

Answer: [N/A]

Justification: The paper does not involve research with human subjects or crowdsourcing. All experiments are conducted on simulated environments and existing animal behavior datasets, and therefore IRB approval is not applicable.

Guidelines:

- The answer [N/A] means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.

16. **Declaration of LLM usage**

Question: Does the paper describe the usage of LLMs if it is an important, original, or non-standard component of the core methods in this research? Note that if the LLM is used only for writing, editing, or formatting purposes and does *not* impact the core methodology, scientific rigor, or originality of the research, declaration is not required.

Answer: [N/A]

Justification: The core methodology of this work does not involve LLMs. While a language model was used only for writing assistance (e.g., grammar and clarity), it did not contribute to the scientific content, model design, or experimental results.

Guidelines:

- The answer [N/A] means that the core method development in this research does not involve LLMs as any important, original, or non-standard components.
- Please refer to our LLM policy in the NeurIPS handbook for what should or should not be described.