Learning Adaptive Diffusion Policies for Hybrid Dynamical Systems

Leroy D'Souza[1*], Akash Karthikeyan[1*], Yash Vardhan Pant[1], Sebastian Fischmeister[1*†‡]

Abstract

Hybrid dynamical systems result from the combination of a set of continuous-variable with a discrete-event system that can be influenced by the *environment* and encompasses systems like legged robots and aircrafts. Model-based control of these systems is challenging, particularly when the systems are complex (high-dimensional non-linear dynamics) and the switching conditions are not perfectly known. We propose a model-free actor-critic learning framework, MoE-Diff, that leverages energy-diffusion for multi-modal action generation with contrastive loss enabling scalable, compositional, and robust decision-making across complex tasks. We use MoE-Diff for control of hybrid systems in apriori unknown environments. Qualitative and ablation studies attribute this improvement to MoE-Diff's ability to learn interpolatable representations and distinct behavior modes across experts enabling compositional generalization and adaptation to unseen scenarios.

1 Introduction

Reinforcement Learning (RL) algorithms are typically developed under the assumption of continuous system dynamics that are invariant to the environment that a system is operating in. However, many systems exhibit hybrid dynamics, i.e., where there is switch in the mode of the governing dynamics due to the operating environment (e.g., a race-car on a partially wet track) or due to the system's configuration itself (e.g., a legged robot when a leg contacts the ground [24], or a powertrain [4]). The problem of controlling such hybrid systems is well studied for low-dimensional piecewise affine systems [18,19], but remains a challenge for general nonlinear systems, particularly in settings where the deployment environment is apriori unknown.

Optimization-based Model Predictive Control (MPC) techniques have been applied to such systems, where discrete mode switches result in mixed-integer formulations that are often intractable without relaxations [5, 22]. They also scale poorly to higher-dimensional systems and require expert knowledge for accurate model design. The latter is particularly challenging for novel environments since it requires knowing where mode switches can occur. While some recent work explores learning system models from data [14], hybrid systems remain largely underexplored in this context.

Learning based approaches enable flexible modeling of dynamics by learning optimal policies directly from data, without explicit system models, and have shown success in adaptive, robust control of complex systems [1, 17]. However, its effectiveness relies on the diversity of experiences in the training data. This translates to observing transitions between modes of the hybrid system. Without such data, learned policies may perform well in isolated modes but fail catastrophically at or close to mode boundaries, especially when transition regions are not known in advance.

Example 1 *Consider an autonomous vehicle navigating in an environment with varying terrain types—such as dry asphalt, wet roads, and snow. Each surface induces different dynamics due to varying friction, corresponding to distinct operational modes. As the vehicle transitions between these terrains, its active dynamics mode changes, yielding a hybrid system. Safe, real-time control of these systems remains a challenge as shown in Figure 1 where the trajectory in blue illustrates a policy trained to perform well on asphalt and ice individually that fails to handle transitions between them and skids off the track. In contrast, a policy trained in hybrid environments can better anticipate and adapt to mode switches. The trajectory in red slows down into the corner and accelerates out of it without crashing.*

Contributions of this work. To address these challenges, we propose MoE-Diff, a model-free online RL framework. Our key contributions are as follows:

▷ MoE-Diff learns an energy-diffusion-based policy repre-

[*1]Department of Electrical and Computer Engineering, University of Waterloo, Waterloo, Canada. l8dsouza@uwaterloo.ca, yash.pant@uwaterloo.ca.

[†*]The authors contributed equally.

[‡]This work was supported in part by Magna International and the NSERC Discovery Grant.

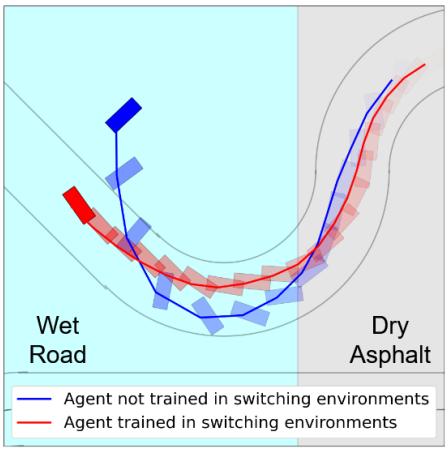

Figure 1: Vehicle trajectories (right to left) in a hybrid environment with switches between different surfaces. The blue trajectory illustrates the difficulty of controlling hybrid systems without mode switching awareness.

sentation that captures multi-modal and interpolatable behaviors. This enables smooth adaptation across hybrid dynamic modes by supporting dynamic switching between discontinuous control strategies.

▷ We parameterize the actor as a Mixture of Experts (MoE), that allows learning of a set of experts, continuous sub-policies, and an interpretable routing mechanism that selects expert(s) based on the input state. Each expert specializes in a distinct mode of behavior, supporting modular control over hybrid dynamics.

▷ MoE-Diff enables end-to-end learning and frames control of hybrid systems as iterative energy minimization, allowing the composition and interpolation of dynamic modes to adapt in apriori unknown environments.

We evaluate our approach through extensive simulations in racing environments [23]. MoE-Diff demonstrates performance on par with the baseline methods on standard tasks with slight differences to training. It also significantly outperforms the baselines in zero-shot generalization tasks to out-of-distribution settings. In particular, we evaluate policies in novel racetrack configurations where MoE-Diff consistently succeeds. We also demonstrate the significance of training policies in hybrid settings, to improve implicitly balancing reward maximization with conservative behavior under mode uncertainty and quantitatively show how this leads to general improvements in performance.

2 Related Works

We briefly cover existing literature on the control of hybrid systems, dividing it into two categories: one for control-theoretic methods and the other for reinforcement learning-based techniques.

2.1 Model-based Control of Hybrid Systems

Most recent model-based control-theoretic work on hybrid systems leverages optimization-based model predictive control (MPC). Learning-based approaches have gained traction in MPC for complex nonlinear systems by improving predictive accuracy through Bayesian models (e.g., Gaussian Processes, Bayesian Neural Networks), which naturally capture uncertainty and enable probabilistic safety constraints [14,37]. Safe model learning for safety-critical systems has also been explored, though under the assumption of uni-modal system dynamics [3, 21].

Introducing discrete variables in MPC results in mixed-integer optimization problems [32], which are computationally challenging for nonlinear systems [6, 10] due to poor scaling with larger optimization horizons which are necessary for performance. Heuristic approximations are commonly used to mitigate this [33]. In cases where the hybrid models are unknown, unsupervised clustering has been applied to learn piecewise linear hybrid models [2, 11].

2.2 Reinforcement Learning for Control of Hybrid Systems

Recent work uses deep learning to learn control barrier functions for stability and safety [38], with some enabling smooth transitions between hybrid modes [27]. Reinforcement learning has also been applied to hybrid system control [25, 29], though often without emphasizing mode transitions driven by environmental changes, a key aspect highlighted in Figure 1.

While model-based approaches have traditionally been used for hybrid systems, *model-free RL* provides an alternative by learning directly from interaction, without requiring system identification or explicit mode-switching logic. However, this poses a significant challenge: the policy must infer and adapt to different dynamic regimes on the fly, while maintaining stability and robustness across discontinuous transitions.

Generative Models for Policy Learning. Recent work has explored the use of diffusion models for behavioral cloning and offline RL [1, 7, 17, 39], demonstrating their effectiveness in capturing multi-modal behavior, modeling discontinuities, and composing skills. These models have shown strong performance in domains with complex dynamics, such as legged locomotion [16] and autonomous driving [43], where traditional unimodal policies struggle to generalize. Building on these advances, energy-based formulations extend diffusion models by incorporating task-aligned objectives through learned energy functions [9, 20], enabling improved generalization to out-of-distribution (OOD) scenarios and better multi-task adaptation. By coupling contrastive learning with diffusion, they provide structured, interpretable policies that score trajectories based on semantic or reward-driven criteria. Despite these advances, most existing methods work in the offline RL setting, where policies are trained on static datasets and lack the ability to explore or adapt to novel scenarios. This limits their effectiveness in dynamic environments, motivating the need for online RL to enable continual learning and policy refinement through interaction.

DIPO [41] introduces a policy improvement method for diffusion-based agents without backpropagating through the full diffusion chain, but its gradient-based updates may drift from the behavior policy, limiting exploration. QSM [30] uses score-matching objectives, which are sensitive to errors in the value function. QVPO [8] builds on DIPO [41] by proposing a Q-weighted diffusion loss, but remains susceptible to mode collapse.

In contrast, we propose a model-free, energy-diffusion framework for online RL that enables multi-modal control and adaptation to unseen scenarios. This is particularly important for hybrid dynamics, where policies must interpolate between previously encountered modes and produce smooth transitions across discontinuous modes.

3 Problem Statement and Preliminaries

We begin by introducing the class of systems and the problem statement addressed in Section 3.1, followed by background for our proposed solution in Section 3.2.

3.1 Hybrid Systems

We consider a hybrid system with state $s_t \in \mathcal{S} \subset \mathbb{R}^n$, control input $a_t \in \mathcal{A} \subset \mathbb{R}^m$, and discrete-time dynamics of the form:

$$s_{t+1} = \sum_{m=1}^{M} \delta^m(s_t, a_t)\left(f^m(s_t, a_t; \mu_m) + w_t^m\right) \quad (1)$$

Here, $\{f^m : \mathcal{S} \times \mathcal{A} \to \mathcal{S}\}_{m=1}^M$ are mode-specific dynamics models, with each being characterized by a particular (likely latent) parameter vector μ_m. w_t^m denotes mode-specific process noise. The indicators $\delta^m(s_t, a_t) : \mathcal{S} \times \mathcal{A} \to \{0, 1\}$ select the active mode at time t. Since only one mode can be active at a time, the domain $\mathcal{S} \times \mathcal{A}$ is partitioned into mutually exclusive regions i.e., $\sum_{m=1}^M \delta^m(s_t, a_t) = 1$. We drop the arguments of δ^m henceforth for brevity.

This formulation models systems whose behavior switches due to environmental variation or internal configuration. For instance, terrain-driven transitions (Example 1) depend on position, while legged robots may switch dynamics based on contact configurations realized by different gaits. In either case, switching is often governed by latent variables, posing challenges for control. The dynamics equation 1 can also be viewed switches between parametrized MDPs, $\mathcal{M}_{\text{set}} = \{(\mathcal{S}, \mathcal{A}, T_{\mu_m}, R, \gamma)\}_{m=1}^M$ where $T_{\mu_m}(\cdot) = f^m(\cdot; \mu_m)$. The switching MDP is defined as,

$$\mathcal{M}_\delta = (\mathcal{M}_{\text{set}}, T_\delta), \quad T_\delta = \sum_{m=1}^{M} \delta^m T_{\mu_m} \quad (2)$$

Here, $R : \mathcal{S} \times \mathcal{A} \to [r_{\min}, r_{\max}]$ is the bounded reward function and $\gamma \in [0, 1]$ is the discount factor. Policies define probability densities $\pi(a \mid s) : \mathcal{S} \times \mathcal{A} \to [0, \infty)$, from which actions are sampled. A trajectory $\tau = \{(s_t, a_t, s_{t+1})\}_{t=0}^{T-1}$ is generated under π via transitions defined by T_δ where $a_t \sim \pi(a \mid s_t)$.

Problem 1 *Given a system subject to the switching MDP defined in equation 2, we aim to learn a control policy $\pi_\theta(a \mid s)$ online that maximizes the expected return $J(\pi) = \mathbb{E}_{\tau\sim\pi}[\sum_{t=0} \gamma^t r_t]$ from interactions with environments where $\{\delta^m\}_{m=1}^M$ are known. When deployed to unseen environments, these policies must control the hybrid system even when the switching functions $\{\delta^m\}_{m=1}^M$ are a priori unknown.*

3.2 Denoising Diffusion Probabilistic Models

Denoising Diffusion Probabilistic Models [15, 28, 36] is a class of generative models that transforms a data distribution into an isotropic Gaussian distribution by adding noise (forward process) and learns to denoise it using a neural network (reverse process), referred to as the denoiser. For the forward process, given a dataset $\mathcal{D}$ of actions corresponding to various behavior modes, let $\mathbf{x}^0 \sim q(\mathbf{x}^0)$ denote the data distribution, where the superscript indicates the diffusion timestep $k \in \{0, \ldots, K\}$. The subscript t corresponds to the MDP timestep. We use bold notation (e.g., $\mathbf{x}$) to represent a sequence or batch. In forward process, Gaussian noise is incrementally added to the data point $\mathbf{x}^0$ using a variance schedule β_k across K timesteps, resulting in a continuous Markov chain:

$$q(\mathbf{x}^k \mid \mathbf{x}^{k-1}) := \mathcal{N}(\mathbf{x}^k; \sqrt{1-\beta_k}\mathbf{x}^{k-1}, \beta_k \mathbf{I}), \quad (3a)$$

$$q(\mathbf{x}^k \mid \mathbf{x}^0) = \mathcal{N}(\mathbf{x}^k; \sqrt{\bar{\alpha}_k}\mathbf{x}^0, (1-\bar{\alpha}_k)\mathbf{I}). \quad (3b)$$

By the Markov property of the forward process, the marginal distribution of the noisy sample at diffusion timestep k can be expressed as a conditional of the data point $q(\mathbf{x}^k \mid \mathbf{x}^0)$, where $\alpha_k = 1 - \beta_k$ and $\bar{\alpha}_k = \prod_{s=0}^k \alpha_s$. Eventually, as $K \to \infty$, $\mathbf{x}^k$ converges to an isotropic Gaussian distribution. During the reverse process, Denoising Diffusion Probabilistic Models (DDPM) samples from the prior distribution $p(\mathbf{x}^k) = \mathcal{N}(\mathbf{x}^k; 0, \mathbf{I})$ and then iteratively denoises it using a learned model $p_\theta(\mathbf{x}^{k-1} \mid \mathbf{x}^k) = \mathcal{N}(\mathbf{x}^{k-1}; \mu_\theta(\mathbf{x}^k, k), \Sigma_\theta(\mathbf{x}^k, t))$, which aims to approximate $q(\mathbf{x}^{k-1} \mid \mathbf{x}^k, \mathbf{x}^0)$, where θ denotes the learnable parameters. The training objective is to minimize the variational lower bound (VLB): $L_{\text{VLB}} = \mathbb{E}_{q(\mathbf{x}^{0:K})}\left[\log \frac{p_\theta(\mathbf{x}^{0:K})}{q(\mathbf{x}^{1:K}|\mathbf{x}^0)}\right]$, which can be simplified, following [15], to the loss function as follows, where $\mathcal{U}$ denotes uniform sampling over timesteps:

$$\mathbb{E}_{k\sim\mathcal{U}[1,K], x^0\sim\mathcal{D}, \epsilon\sim\mathcal{N}(0,I)}\left[||\epsilon - \epsilon_\theta(\sqrt{\bar{\alpha}_k}x^0 + \sqrt{1-\bar{\alpha}_k}\epsilon, k)||^2\right]. \quad (4)$$

4 MoE-Diff: Method

We aim to learn a policy to generalize to *unseen hybrid scenarios* as described in Problem 1 and Example 1. In this section, we introduce MoE-Diff, which builds on the following key components: (i) learning to model different dynamic modes in the environment (which are unknown *a priori*), (ii) preventing mode collapse during training, a common issue in RL where modes associated with higher

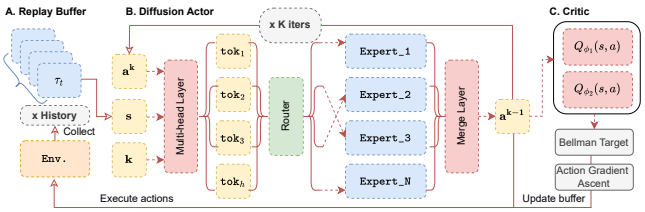

Figure 2: **MoE-Diff Overview. A.** A replay buffer stores past observations collected from environment interactions. **B.** The Diffusion Actor, parameterized as a MH-MoE [40], takes a noisy action input $\boldsymbol{a}^k$ and iteratively denoises it to produce $\boldsymbol{a}^0$, conditioned on the state and diffusion timestep; this final action is executed in the environment. **C.** The Critic, following [13], estimates Bellman residuals, and the action gradient $\nabla_a Q_\phi(s,a)$ is used to refine actions during policy improvement. The refined action is then appended back to the buffer.

Q-values dominate the learned policy, and (iii) learning an interpretable policy that allows flexible composition of dynamic modes to sample actions that can adapt to novel hybrid scenarios. An overview of the method is provided in Figure 2, and the algorithm is detailed in Algorithm 1.

4.1 Energy-Diffusion Policy for Hybrid Dynamics

Policy Representation. To model complex multi-modal dynamics, we parameterize the RL policy as the reverse process of diffusion model: $\pi_\theta(\boldsymbol{a}\,|\,\boldsymbol{s}) = p_\theta(\boldsymbol{a}^{0:N}\,|\,\boldsymbol{s})$ [15, 39], where the final action from the reverse sampling chain is used for RL evaluation. The reverse process is iterative, transforming samples drawn from an isotropic Gaussian distribution into the action distribution conditioned on the state.

$$\boldsymbol{a}^{k-1}\,|\,\boldsymbol{a}^k = \frac{\boldsymbol{a}^k}{\sqrt{\alpha_k}} - \frac{\beta_k}{\sqrt{\alpha_k(1-\bar{\alpha}_k)}}\boldsymbol{\epsilon}_\theta(\boldsymbol{a}^k, \boldsymbol{s}, k) + \sqrt{\beta_k}\boldsymbol{\epsilon}, \quad (5a)$$

$$\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I}), \quad \text{for } k = K, \ldots, 1. \quad (5b)$$

Policy Learning. For the policy improvement step, instead of injecting Q-value function guidance directly into the reverse diffusion process [39], which requires backpropagating through the entire diffusion chain. We follow the approach in [41] and adopt action gradient ascent for policy improvement, as shown in line 14 of Algorithm 1. We use the double Q-learning strategy from [13] to obtain Q-value estimates. This forms the basis of our policy improvement step, coupled with a conditional diffusion loss modified from equation 4. Although a Q-weighted update rule can be used [8], our experiments show that it leads to unstable training and more susceptible to mode collapse.

$$\mathcal{L}_D(\theta) = \mathbb{E}\left[\|\epsilon - \pi_\theta(\sqrt{\bar{\alpha}_k}a^0 + \sqrt{1-\bar{\alpha}_k}\epsilon, s, k)\|^2\right] \quad (6)$$

where the expectation is computed over $s, a^0 \sim \mathcal{D}$, $\epsilon \sim \mathcal{N}(0, I)$, $k \sim \mathcal{U}[1, K]$.

Algorithm 1 MoE-Diff- Training

1: **Initialize** Diffusion actor π_θ, critics Q_{ϕ_1}, Q_{ϕ_2}, target networks $\pi_{\theta'}, Q_{\phi_1'}, Q_{\phi_2'}$, buffer $\mathcal{D} \leftarrow \emptyset$, Set noise schedule $\{\alpha_k\}_{k=1}^K$ with $\bar{\alpha}_k = \prod_{j=1}^k \alpha_j$, action update size η
2: **for** iteration $e = 1, \ldots, E$ **do**
3: **for** timestep $t = 0, \ldots, T$ **do**
4: Observe state s_t and sample actions iteratively: $a_t^0 \sim \pi_\theta(\cdot \mid s_t)$ ▷ by Eq. 5
5: Reward: $r_t = R(s_t, a_t)$ ▷ optionally $a_t' = a_t + \epsilon; \epsilon \sim \mathcal{N}(0, I)$
6: Store (s_t, a_t, s_{t+1}, r_t) in buffer $\mathcal{D}$
7: **end for**
8: **for** update step $g = 1, \ldots, G$ **do**
9: Sample batch $(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{r}, \boldsymbol{s'})$ from replay buffer
10: Update parameters ϕ following [13] ▷ Train Q-function
11: Update parameters θ using the losses in equation 6, 7 ▷ Train Diffusion Policy
12: **for** $c = 1, \ldots, C$ **do**
13: $a^{0,c+1} \leftarrow a^{0,c} + \eta \nabla_a Q_\phi(s,a)\big|_{a=a^{0,c}}$ ▷ Q-guided Action Refinement
14: Add refined action to buffer: $\mathcal{D} \leftarrow \mathcal{D} \cup \{(s, a^{0,C})\}$
15: **end for**
16: **end for**
17: **end for**

Contrastive Loss. While DDPM enables learning of expressive multi-modal action distributions, it remains limited in its ability to generalize to unseen scenarios, particularly those that require switching and interpolation between previously observed dynamics modes. To address this, we reframe the problem of modeling hybrid dynamics as an *energy minimization task*, where the policy iteratively recovers optimal actions by sampling from a learned energy landscape that captures generalizable control behavior. We define the energy of a state-action pair at diffusion timestep k as a scalar value, computed by summing over the relevant dimensions of the learned energy vector: $E_\theta = \sum_{i=1}^{d_{\text{dim}}} [\pi_\theta(\boldsymbol{a}, \boldsymbol{s}, k)]_i$ where d_{dim} denotes the dimensionality of the action considered for energy evaluation. This formulation encourages the model to assign low energy (high-likelihood) to ground-truth actions and higher energy to corrupted (i.e., unsafe) alternatives. To learn global energy landscape, we introduce a contrastive loss that penalizes cases where corrupted actions receive lower energy than the true ones. To generate negative samples, we perturb the ground-truth action using additive Gaussian noise: $\boldsymbol{a}^- = \boldsymbol{a}^* + \lambda\epsilon$, where λ is a hyperparameter and $\epsilon \sim \mathcal{N}(0, \mathbf{I})$. The contrastive loss is then defined as:

$$\mathcal{L}_{\text{C}}(\theta) = -\log\left(\frac{e^{-\pi_\theta(\boldsymbol{x}, \boldsymbol{a}^*, k)}}{e^{-\pi_\theta(\boldsymbol{x}, \boldsymbol{a}^*, k)} + e^{-\pi_\theta(\boldsymbol{x}, \boldsymbol{a}^-, k)}}\right). \quad (7)$$

As a result, the policy learns smooth and interpolatable action representations across diverse hybrid dynamics

modes.

4.2 Key Design Decisions

To model multi-modal and hybrid dynamics with interpretable and interpolatable behavior, we parameterize the actor using a Multi-Head Mixture-of-Experts (MH-MoE) network. MH-MoE [40] enables finer-grained control and better task adaptation by a multi-head mechanism for generating sub-tokens and routing them independently through sparsely activated experts. This promotes expert specialization and efficient utilization.

Token Routing via Softmax Gating. Each input embedding is split into multiple sub-tokens, which are independently routed to a pool of experts. A softmax gating mechanism is used to assign routing weights based on the similarity between sub-token representations and learned expert-specific routing keys. To promote better expert utilization and avoid collapse into a few dominant experts, we adopt *noisy top-k routing* [35], where small Gaussian noise is added to the gating logits prior to selecting the top-k experts for each sub-token.

Router z-Loss for Stability. To prevent peaked expert distributions and stabilize training, we incorporate the *router z-loss* [44], a regularization term defined as $\mathcal{L}_{\mathrm{Z}} = \lambda_z \cdot \mathbb{E}[(\log \sum_{p=1}^{N} \exp(l_p))^2]$, where l_p are the gating logits over experts and λ_z is a small regularization coefficient. This encourages smoother expert assignments and mitigates mode collapse in the routing mechanism.

Training Objective. The overall training objective involves alternating updates between the actor and critic networks, with action gradient ascent steps as shown in lines 9–14 of Algorithm 1. Specifically, the actor is trained using the combined loss: $\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{D}} + \mathcal{L}_{\mathrm{C}} + \mathcal{L}_{\mathrm{Z}}$, where $\mathcal{L}_{\mathrm{D}}$ is the diffusion loss, $\mathcal{L}_{\mathrm{C}}$ is the contrastive loss, and $\mathcal{L}_{\mathrm{Z}}$ is the router z-loss regularizer. The critic is updated according to the Bellman residual using the double Q-learning formulation [13].

5 Simulation Studies and Results

We aim to answer the following questions through our simulation studies,

▷ **Generalization under hybrid dynamics.** How well do MoE-Diff and baseline methods generalize across increasing levels of hybrid complexity, ranging from unseen single-mode settings to fully hybrid rollouts?

▷ **Multi-Modality.** Do the proposed models scale effectively to complex hybrid rollouts and exhibit multimodal behavior under diverse switching patterns?

▷ **Implicit dynamic awareness through hybrid training.** Does training in hybrid environments enable the policy to implicitly recognize and adapt to changes in dynamics, even without explicit friction information?

▷ **Zero-Shot Generalization to Novel Tasks.** Can hybrid-trained policies generalize zero-shot to novel tasks, such as navigating unseen racetracks or adapting online to new obstacles like wedges?

To address the above questions, we evaluate our model on three challenging environments with hybrid system dynamics a racetrack environment. All experiments are implemented in Python 3.8 and executed on a 12-core CPU with an RTX A6000 GPU.

5.1 Baselines

UP-True [42]: An algorithm that attempts to learn a *universal policy* and receives ground truth information of the parametric assignments and switching functions in equation 1. While the original paper uses TRPO, we re-implement the algorithm with SAC [12] since it yields better results

SAC [12, 31]: is an off-policy actor-critic algorithm based on the maximum entropy reinforcement learning framework. In this framework, the actor aims to simultaneously maximize the expected return and the policy entropy.

PPO [31, 34]: An on-policy policy gradient technique that improves stability during training policy updates by maximizing a clipped surrogate objective that prevents significant changes in the policy between successive updates.

SAC-Switching: Here, we train *separate* policies corresponding to different coefficients of friction. This can be seen as training an expert for each friction coefficient and switching between them. This cannot generalize to environments with unseen friction coefficients. We also aim to demonstrate that this fails to generalize to hybrid settings, validating the benefits of the MoE component in the architecture.

SAC-MoE: A variant of SAC where the actor network is parameterized as a MoE instead of an MLP, enabling the learning of a set of experts, continuous sub-policies that are composed via a learned router to handle hybrid dynamics. The module is trained in an end-to-end fashion.

DiPo $\oplus$ $\mathcal{C}$: A **Di**ffusion **Po**licy variant of MoE-Diff where the actor in Algorithm 1 is parameterized with an MLP instead of a MoE. Note that this variant uses the losses defined in equations 6 and 7. .

UP-True and SAC-Switching are provided with ground-truth knowledge of where the friction changes occur even in deployment settings. Since UP-True also has ground-truth friction knowledge during training, this baseline is treated as an oracle. SAC, SAC-MoE and PPO do not observe friction values during training. DiPo $\oplus$ $\mathcal{C}$ and MoE-Diff learn to estimate friction values over the course of training.

5.2 Racetrack Results

Here, we evaluate policies in a racing environment [23] where agents are rewarded for maintaining speeds close to 25 m/s. Hybrid dynamics arise due to spatially varying friction along the racetrack, especially affecting behavior at high speeds and sharp turns. The control policy must learn to adapt to low-friction regions and demonstrate caution around corners, without prior knowledge of mode switch locations. Obstacles line the track boundaries, and episodes terminate upon collision. The reward includes speed and track progress components, with penalties for collisions and excessive actions.

Friction setup	Policy			
	SAC	SAC-Switching	UP-True	MoE-Diff
0.5	98.65 (89.67)	**313.16 (55.56)**	223.81 (99.53)	207.78 (1.32)
0.4	68.87 (62.60)	NA	**225.32 (80.87)**	204.73 (0.77)
0.3	28.91 (16.25)	**239.99 (62.67)**	109.29 (76.97)	150.48 (78.37)
0.25	15.14 (9.08)	NA	**34.43 (26.49)**	23.30 (23.08)
[1.0, 0.5]	30.44 (21.96)	37.54 (19.59)	45.89 (26.20)	**207.48 (1.26)**
[0.3, 0.5]	39.44 (21.04)	140.48 (98.51)	65.44 (46.97)	**205.72 (0.83)**
[0.25, 0.4]	39.94 (20.23)	NA	43.45 (23.79)	**168.75 (65.21)**

Table 1: **Performance of single friction environment-trained policies.** Policies are evaluated on single friction environments in the first half, and on hybrid friction racetracks in the second half.

Policy	Single Friction Env.		Hybrid Friction Env.	
	0.5	0.3	[1.0, 0.5]	[0.3, 0.5]
SAC-A	**274.95 (36.56)**	33.30 (21.35)	**268.36 (69.70)**	40.82 (18.30)
SAC-B	243.73 (79.98)	18.89 (9.34)	197.31 (102.44)	34.09 (17.77)
SAC-C	257.58 (59.65)	**103.26 (75.60)**	242.24 (77.06)	**180.70 (103.45)**
UP-True-A	222.64 (96.17)	37.80 (20.33)	120.09 (84.90)	43.99 (18.16)
UP-True-B	150.14 (110.17)	29.37 (20.76)	68.84 (51.33)	36.00 (17.16)
UP-True-C	**271.71 (16.08)**	**87.07 (62.88)**	**277.65 (36.93)**	**66.59 (34.70)**

Table 2: **Importance of training in *informative* hybrid environments.** Policies are learnt in hybrid environments. "SAC-A" denotes SAC trained in environments generated using hybrid friction environment generation approach A as described in Section 5.2.

Friction setup	Evaluation Setting 1					Evaluation Setting 2					
	SAC	UP-True	SAC-MoE	DiPo $\oplus$ $\mathcal{C}$	MoE-Diff	SAC	UP-True	SAC-MoE	DiPo $\oplus$ $\mathcal{C}$	MoE-Diff (Single)	MoE-Diff
0.5	257.6 (59.7)	271.7 (16.1)	**273.3 (1.7)**	211.2 (1.7)	246.2 (2.0)	104.9 (80.3)	111.2 (69.0)	224.0 (52.2)	158.1 (83.8)	196.0 (40.4)	**224.4 (69.1)**
0.4	246.2 (61.8)	**263.5 (22.3)**	239.8 (57.7)	208.6 (1.6)	246.1 (2.2)	58.8 (54.4)	105.9 (66.9)	**218.0 (51.0)**	116.3 (76.4)	195.0 (27.1)	162.3 (85.6)
0.3	103.3 (75.6)	87.1 (62.9)	109.4 (65.8)	**176.3 (59.8)**	95.2 (57.5)	26.2 (17.1)	81.1 (55.2)	182.2 (62.4)	44.8 (36.3)	**192.0 (2.5)**	75.0 (52.5)
[1.0, 0.5]	242.2 (77.1)	277.6 (36.9)	**279.9 (1.8)**	216.6 (1.9)	252.0 (1.4)	176.7 (100.9)	142.7 (82.5)	224.9 (52.8)	160.9 (89.0)	194.4 (40.1)	**232.8 (58.4)**
[0.3, 0.5]	180.7 (103.4)	66.6 (34.7)	155.2 (80.0)	**208.3 (1.98)**	123.5 (75.4)	49.6 (44.5)	100.2 (66.3)	211.5 (59.5)	120.1 (82.4)	192.6 (39.7)	**233.6 (51.6)**
[0.25, 0.4]	29.1 (17.0)	39.5 (18.5)	34.8 (17.6)	**191.4 (43.7)**	33.5 (18.4)	29.5 (37.3)	95.1 (60.6)	**189.9 (70.4)**	88.8 (71.4)	189.7 (26.4)	144.2 (78.2)

Table 3: **Performance of hybrid friction environment-trained policies.** Each policy is evaluated under two evaluation settings. The top and bottom halves evaluate single and hybrid-friction tracks respectively.

We compare policies trained under two environment configurations,

Single Friction Environments: The entire track uses a single friction coefficient per episode. After each episode, a new coefficient is picked to ensure balanced sampling across $\mu \in \{1.0, 0.5, 0.3\}$, preventing any one value from dominating the dataset. No mode transitions are observed in training.

Hybrid Friction Environments: Friction varies across the track, forming a switching MDP as defined in equation 2. Friction transitions are placed at turns (as visualized in Figure 3, leveraging the knowledge that high speed and steering angle induce significant lateral friction forces. We use two friction coefficients per environment: one active at turns and the other active everywhere else. Hence, each hybrid setup is defined by a *coefficient pair*.

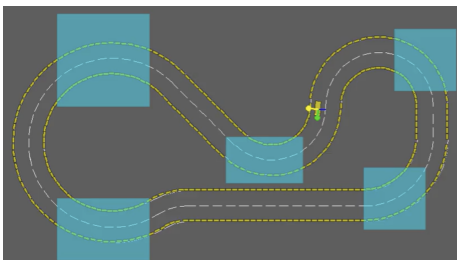

Figure 3: Visualizing of the hybrid training environment. The blue patches have one friction coefficient while the rest of the track has a different coefficient.

All agents are trained on $\mu \in \{1.0, 0.5, 0.3\}$ and evaluated on their ability to generalize to unseen friction values and layouts in two scenarios:

Evaluation Setting 1: Same track layout as training, but with friction transitions at previously unseen locations.

Evaluation Setting 2: A completely new track layout with friction transitions unseen during training.

We first evaluate policies trained in single friction environments under Evaluation Setting 1. As shown in the first half of Table 1, these policies perform well in single friction environments. SAC-Switching can be seen as learning separate expert policies for $\mu \in \{0.5, 0.4, 0.3\}$. When provided with ground truth knowledge of δ^m equation 1, it is no surprise that this policy performs best as it simply picks the expert corresponding to a fixed value of μ for the entire trajectory. However, this approach, along with the other baselines still suffers from a lack of ability to zero-shot generalize to hybrid environments as seen from the second half of the table, often crashing at turns where friction coefficient switches occur. In contrast, MoE-Diff demonstrates reasonably good performance with significantly less variance compared to the baselines on single friction tracks with $\mu \in \{0.5, 0.4\}$. It also retains performance when deployed to unseen hybrid environments even though these switching transitions are not observed in its training dataset. We now consider training in hybrid friction environments. Designing such environments during training is non-trivial due to the need to balance data collection across coefficient pairs and avoid overfitting to easier settings. After each episode, we must choose which coefficient pair to sample next. We evaluate three strategies,

A: Maintain balanced sampling across all coefficients, similar to the single friction setup.

B: Cycle through all coefficient pairs in a fixed order.

C: With probability $p = 0.95$, select the coefficient pair with the lowest average episode reward (over a moving window of size 10). With probability $1 - p$, sample a coefficient pair from the distribution characterized by how many episodes it has been since each coefficient pair was last used.

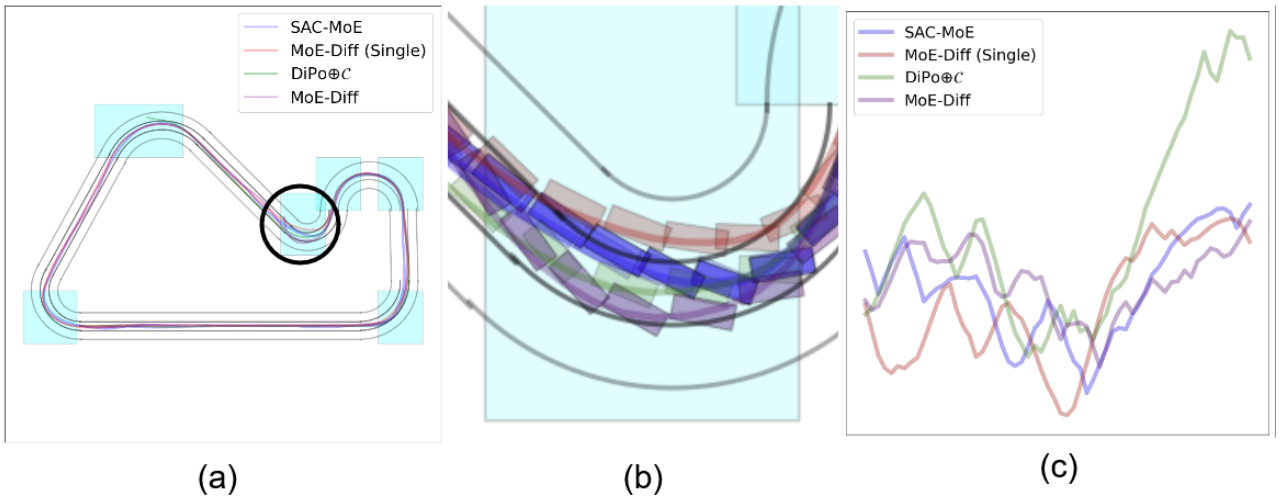

Figure 4: (a) Trajectories plots for different policies on a zero-shot generalization task to a previously unseen racetrack layout with friction changes. (b) Visualization of vehicle boxes over a sharp turn section of the track. (c) Velocity plots corresponding to the turn in (b) across different policies.

Table 2 compares SAC and UP-True policy performance using each strategy. Approach C generally outperforms the others across both single and hybrid test environments, as it balances targeted exploration of challenging coefficient pairs with coverage of the full set of coefficient pairs somewhat similar to active domain randomization [26]. Approach A learns a policy that is not conducive to low friction coefficients hence performing exceeding well on higher friction coefficients but failing in lower friction environments. This is because driving lower friction coefficients can be seen as harder tasks and simply maintaining a balance between all friction coefficients in the dataset results in a policy that converges to good performance on the easier tasks, i.e., higher friction coefficients. All subsequent hybrid training results use approach C.

We evaluate these policies on both the evaluation settings previously described. Table 3 shows that hybrid training improves performance in evaluation setting 1 across most models in both single and hybrid evaluation environments when compared against the values reported in Table 1. Samples that reflect transitions between friction values seem to allow the model to get a better understanding of the hybrid dynamics of the system. The table also shows that the models that learn to estimate friction coefficients i.e., SAC-MoE, DiPo $\oplus\, \mathcal{C}$ and MoE-Diff are capable of generalizing to unseen settings, better than even UP-True which has ground truth friction knowledge. In this setting, there is no one model that performs better than the others consistently. MoE-Diff and SAC-MoE show the ability to generalize to previously the previous unseen racetrack layout with DiPo $\oplus\, \mathcal{C}$ not being as conservative as it should. In contrast, MoE-Diff trained in single environments demonstrates remarkable consistency across the rewards reported for different settings which is due to the conservative policy it learns.

A visualization of the resulting trajectories for a subset of policies that perform the best in evaluation setting to for a hybrid friction environment [0.3, 0.5] is visualized in Figure 4. We see that MoE-Diff trained in single friction environments lags behinds the others slightly and exhibits the largest dip in velocity when making the turn which reflects the conservative values previously discussed. In contrast, DiPo $\oplus\, \mathcal{C}$ has velocity higher than the others and accelerates out of the turn the fastest, exhibiting a more aggressive policy leading to an early crash.

6 Conclusion

Summary. In conclusion, we introduce the challenge of controlling hybrid systems where the environment plays a role in determining when the system switches between different dynamics modes. Training in hybrid environments generally improves the performance of policies in unseen hybrid environments, demonstrating the importance of observing transitions between different dynamics modes during training. We further show how using diffusion and mixture-of-expert components in policy learning can significantly improve performance over policies that receive ground truth friction knowledge in new environments. models that showing the import. However, the approach outlined here is not without limitations.

Limitations and Future Work. The curriculum learning approach does not scale with an increase in the number of parametric variables that affect the system dynamics. Also, control over a lookahead horizon could significantly improve performance and reduce conservatism in the policy. Extending the policies to account for such lookahead remains for future work.

References

[1] Anurag Ajay, Yilun Du, Abhi Gupta, Joshua B. Tenenbaum, Tommi S. Jaakkola, and Pulkit Agrawal. Is conditional generative modeling all you need for decision making? In *The Eleventh International Conference on Learning Representations*, 2023.

[2] Alberto Bemporad. A piecewise linear regression and classification algorithm with application to learning and model predictive control of hybrid systems. *IEEE Transactions on Automatic Control*, 2022.

[3] Felix Berkenkamp and Angela P. Schoellig. Safe and robust learning control with Gaussian processes. In *2015 European Control Conference (ECC)*, Linz, Austria, July 2015. IEEE.

[4] Thomas J Böhme and Benjamin Frank. Hybrid systems, optimal control and hybrid vehicles. *Cham, CH: Springer International*, pages 401–428, 2017.

[5] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. *Predictive control for linear and hybrid systems*. Cambridge University Press, 2017.

[6] Martin Buss, Markus Glocker, Michael Hardt, Oskar von Stryk, Roland Bulirsch, and Günther Schmidt. Nonlinear hybrid dynamical systems: Modeling, optimal control, and applications. In *Modelling, Analysis, and Design of Hybrid Systems*, 2002.

[7] Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems*, 2023.

[8] Shutong Ding, Ke Hu, Zhenhao Zhang, Kan Ren, Weinan Zhang, Jingyi Yu, Jingya Wang, and Ye Shi. Diffusion-based reinforcement learning via q-weighted variational policy optimization. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[9] Yilun Du, Jiayuan Mao, and Joshua B. Tenenbaum. Learning iterative reasoning through energy diffusion. In *Forty-first International Conference on Machine Learning*, 2024.

[10] Leroy D'Souza and Yash Vardhan Pant. Stochastic hybrid model predictive control using gaussian processes for systems with piecewise residual dynamics. In *2023 American Control Conference (ACC)*, 2023.

[11] Giancarlo Ferrari-Trecate, Marco Muselli, Diego Liberati, and Manfred Morari. A clustering technique for the identification of piecewise affine systems. *Automatica*, 39(2):205–217, 2003.

[12] Tuomas Haarnoja, Aurick Zhou, Kristian Hartikainen, George Tucker, Sehoon Ha, Jie Tan, Vikash Kumar, Henry Zhu, Abhishek Gupta, Pieter Abbeel, et al. Soft actor-critic algorithms and applications. *arXiv preprint arXiv:1812.05905*, 2018.

[13] Hado Hasselt. Double q-learning. *Advances in Neural Information Processing Systems (NeurIPS)*, 23, 2010.

[14] Lukas Hewing, Juraj Kabzan, and Melanie N. Zeilinger. Cautious model predictive control using gaussian process regression. *IEEE Transactions on Control Systems Technology*, 2020.

[15] Jonathan Ho, Ajay Jain, and Pieter Abbeel. Denoising diffusion probabilistic models. *Advances in neural information processing systems*, 33:6840–6851, 2020.

[16] Xiaoyu Huang, Yufeng Chi, Ruofeng Wang, Zhongyu Li, Xue Bin Peng, Sophia Shao, Borivoje Nikolic, and Koushil Sreenath. Diffuseloco: Real-time legged locomotion control with diffusion from offline datasets. In *8th Annual Conference on Robot Learning*, 2024.

[17] Michael Janner, Yilun Du, Joshua Tenenbaum, and Sergey Levine. Planning with diffusion for flexible behavior synthesis. In *International Conference on Machine Learning*, pages 9902–9915. PMLR, 2022.

[18] Zachary Jarvis-Wloszek, Ryan Feeley, Weehong Tan, Kunpeng Sun, and Andrew Packard. Some controls applications of sum of squares programming. In *42nd IEEE international conference on decision and control (IEEE Cat. No. 03CH37475)*, volume 5, pages 4676–4681. IEEE, 2003.

[19] Mikael K-J Johansson. *Piecewise linear control systems: a computational approach*, volume 284. Springer, 2003.

[20] Akash Karthikeyan and Yash Vardhan Pant. Genplan: Generative sequence models as adaptive planners. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(17):17797–17804, Apr. 2025.

[21] Torsten Koller, Felix Berkenkamp, Matteo Turchetta, and Andreas Krause. Learning-based model predictive control for safe exploration. In *2018 IEEE conference on decision and control (CDC)*. IEEE, 2018.

[22] Mircea Lazar. Model predictive control of hybrid systems: Stability and robustness. 2006.

[23] Edouard Leurent. An environment for autonomous driving decision-making. `https://github.com/eleurent/highway-env`, 2018.

[24] He Li and Patrick M. Wensing. Hybrid systems differential dynamic programming for whole-body motion planning of legged robots. *IEEE Robotics and Automation Letters*, 5:5448–5455, 2020.

[25] Hang Liu, Sangli Teng, Ben Liu, Wei Zhang, and Maani Ghaffari. Discrete-time hybrid automata learning: Legged locomotion meets skateboarding, 2025.

[26] Bhairav Mehta, Manfred Diaz, Florian Golemo, Christopher J Pal, and Liam Paull. Active domain randomization. In *Conference on Robot Learning*, pages 1162–1176. PMLR, 2020.

[27] Yue Meng and Chuchu Fan. Hybrid systems neural control with region-of-attraction planner. In *Learning for Dynamics and Control Conference*, pages 1400–1415. PMLR, 2023.

[28] Alexander Quinn Nichol and Prafulla Dhariwal. Improved denoising diffusion probabilistic models. In *International conference on machine learning*, 2021.

[29] Michael Poli, Stefano Massaroli, Luca Scimeca, Sanghyuk Chun, Seong Joon Oh, Atsushi Yamashita, Hajime Asama, Jinkyoo Park, and Animesh Garg. Neural hybrid automata: Learning dynamics with multiple modes and stochastic transitions. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[30] Michael Psenka, Alejandro Escontrela, Pieter Abbeel, and Yi Ma. Learning a diffusion model policy from rewards via q-score matching. *arXiv preprint arXiv:2312.11752*, 2023.

[31] Antonin Raffin, Ashley Hill, Adam Gleave, Anssi Kanervisto, Maximilian Ernestus, and Noah Dormann. Stable-baselines3: Reliable reinforcement learning implementations. *Journal of Machine Learning Research*, 2021.

[32] A. Richards and J. How. Mixed-integer programming for control. In *Proceedings of the 2005, American Control Conference, 2005.*

[33] Giulio Ripaccioli, Daniele Bernardini, Stefano Di Cairano, Alberto Bemporad, and Ilya V Kolmanovsky. A stochastic model predictive control approach for series hybrid electric vehicle power management. In *Proceedings of the 2010 American control conference*. IEEE, 2010.

[34] John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms, 2017.

[35] Noam Shazeer, Azalia Mirhoseini, Krzysztof Maziarz, Andy Davis, Quoc Le, Geoffrey Hinton, and Jeff Dean. Outrageously large neural networks: The sparsely-gated mixture-of-experts layer. *arXiv preprint arXiv:1701.06538*, 2017.

[36] Jascha Sohl-Dickstein, Eric Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *International conference on machine learning*, pages 2256–2265. PMLR, 2015.

[37] Bart PG Van Parys, Daniel Kuhn, Paul J Goulart, and Manfred Morari. Distributionally robust control of constrained stochastic systems. *IEEE Transactions on Automatic Control*, 2015.

[38] Ruigang Wang and Ian Manchester. Direct parameterization of lipschitz-bounded deep networks. In *International Conference on Machine Learning*, pages 36093–36110. PMLR, 2023.

[39] Zhendong Wang, Jonathan J Hunt, and Mingyuan Zhou. Diffusion policies as an expressive policy class for offline reinforcement learning. In *The Eleventh International Conference on Learning Representations*, 2022.

[40] Xun Wu, Shaohan Huang, Wenhui Wang, Shuming Ma, Li Dong, and Furu Wei. Multi-head mixture-of-experts. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024.

[41] Long Yang, Zhixiong Huang, Fenghao Lei, Yucun Zhong, Yiming Yang, Cong Fang, Shiting Wen, Binbin Zhou, and Zhouchen Lin. Policy representation via diffusion probability model for reinforcement learning. *arXiv preprint arXiv:2305.13122*, 2023.

[42] Wenhao Yu, Jie Tan, C. Karen Liu, and Greg Turk. Preparing for the unknown: Learning a universal policy with online system identification. In *Robotics: Science and Systems XIII, Massachusetts Institute of Technology, Cambridge, Massachusetts, USA, July 12-16, 2017*.

[43] Ziyuan Zhong, Davis Rempe, Danfei Xu, Yuxiao Chen, Sushant Veer, Tong Che, Baishakhi Ray, and Marco Pavone. Guided conditional diffusion for controllable traffic simulation. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 3560–3566, 2023.

[44] Barret Zoph, Irwan Bello, Sameer Kumar, Nan Du, Yanping Huang, Jeff Dean, Noam Shazeer, and William Fedus. St-moe: Designing stable and transferable sparse expert models, 2022.