# Hybrid Recurrent Models Support Emergent Descriptions for Hierarchical Planning and Control

Poppy Collis [* 1]   Ryan Singh [* 1 2]   Paul F Kinghorn [1]   Christopher L Buckley [1 2]

## Abstract

An open problem in artificial intelligence is how systems can flexibly learn discrete abstractions that are useful for solving inherently continuous problems. Previous work has demonstrated that a class of hybrid state-space model known as recurrent switching linear dynamical systems (rSLDS) discover meaningful behavioural units via the piecewise linear decomposition of complex continuous dynamics (Linderman et al., 2016). Furthermore, they model how the underlying continuous states drive these discrete mode switches. We propose that the rich representations formed by an rSLDS can provide useful abstractions for planning and control. We present a novel hierarchical model-based algorithm inspired by Active Inference in which a discrete MDP sits above a low-level linear-quadratic controller. The recurrent transition dynamics learned by the rSLDS allow us to (1) specify temporally-abstracted sub-goals in a method reminiscent of the options framework, (2) lift the exploration into discrete space allowing us to exploit information-theoretic exploration bonuses and (3) 'cache' the approximate solutions to low-level problems in the discrete planner. We successfully apply our model to the sparse Continuous Mountain Car task, demonstrating fast system identification via enhanced exploration and non-trivial planning through the delineation of abstract sub-goals.

## 1. Introduction

In a world that is inherently continuous, the brain's apparent capacity to distil discrete concepts from sensory data repre-

sents a highly desirable feature in the design of autonomous systems. Humans are able to flexibly specify abstract sub-goals during planning, thereby reducing problems into manageable chunks (Newell & Simon, 1972; Gobet et al., 2001). Furthermore, they are able to transfer this knowledge across new tasks; a process which has proven a central challenge in artificial intelligence (d'Avila Garcez & Lamb, 2023). Translating problems into discrete space offers distinct advantages in decision-making. Namely, the computationally feasible application of information-theoretic measures (e.g. information-gain), as well as the direct implementation of classical techniques such as dynamic programming (LaValle, 2006; Friston et al., 2023). One prevalent approach to tackling continuous spaces involves the simple grid-based discretisation of the state-space, however this becomes extremely costly as the dimensionality increases (Coulom, 2007; Mnih et al., 2015). We therefore ask how we might be able to smoothly handle the presence of continuous variables whilst maintaining the benefits of decision-making in the discrete domain.

To address this, we explore the rich representations learned by recurrent switching linear dynamical systems (rSLDS) in the context of planning and control. This class of hybrid state-space model consists of discrete latent states that evolve via Markovian transitions, which act to index a discrete set of linear dynamical systems (Linderman et al., 2016). Importantly, a continuous dependency in the discrete state transition probabilities is included in the generative model. By providing an understanding of the continuous latent causes of switches between discrete modes, this recurrent transition structure can be exploited such that a controller can flexibly specify inputs to drive the system into a desired region of the state-action space. By embracing the established control-theoretic strategy of piecewise linear decomposition of nonlinear dynamics, our approach lies in contrast to the comparatively opaque solutions found by continuous function approximators (Liberzon, 2003; Mnih et al., 2015). Using statistical methods to fit these models provides a means by which we can effectively perform online discovery of useful non-grid discretisations of the state-space for system identification and control.

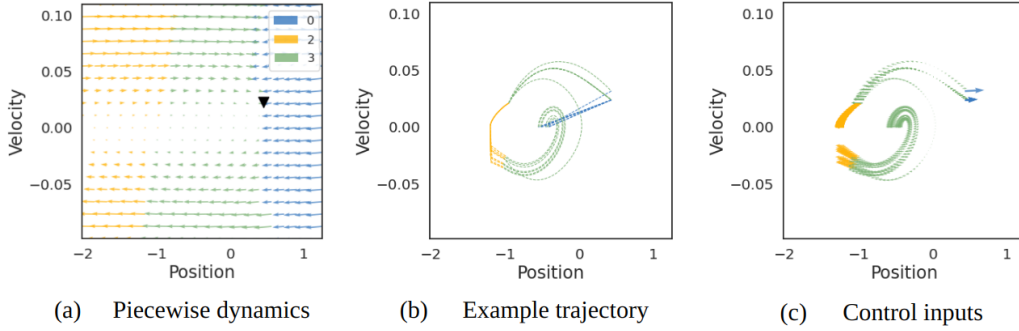We describe a novel model-based algorithm inspired by Ac-

*Equal contribution [1]School of Engineering and Informatics, University of Sussex, Brighton, UK [2]VERSES AI Research Lab, Los Angeles, California, USA. Correspondence to: Poppy Collis <pzc20@sussex.ac.uk>.

(a)  Piecewise dynamics      (b)  Example trajectory      (c)  Control inputs

*Figure 1.* **HHA solves nonlinear problems via specifying abstract sub-goals in state-space.** (a) Piecewise linear dynamics of the Continuous Mountain Car state-space found by rSLDS. Reward location shown (*black triangle*). While the rSLDS retrives 5 modes in total, here we plot only the modes seen in the position-velocity ($x$) space without showing the control input ($u$) axis. (b) Example trajectory (segments coloured by mode) showing the HHA consistently navigating to the goal. (c) Continuous control input (coloured by discrete action specified by planner and arrow size indicating magnitude and direction) over same example trajectory in (b).

tive Inference (Parr et al., 2022), in which a discrete MDP, informed by the representations of an rSLDS, interfaces with a finite horizon linear-quadratic regulator (LQR) implementing closed-loop control. We demonstrate the efficacy of this algorithm by applying it to the classic control task of Continuous Mountain Car (OpenAI, 2021). We show that information-theoretic exploration drive integrated with the emergent piecewise description of the task-space facilitates fast system identification to find successful solutions to this non-trivial planning problem.

### 1.1. Contributions

- The enhancement of planning via the introduction of temporally-abstracted sub-goals by decoupling a discrete MDP from the continuous clock time using the emergent representations from an rSLDS.

- The lifting of information-seeking decision-making into a (discrete) abstraction of the states enabling efficient exploration and thereby reducing sensitivity to the dimensionality of the task-space.

## 2. Related work

In the context of control, hybrid models in the form of piecewise affine (PWA) systems have been rigorously examined and are widely applied in real-world scenarios (Bemporad et al., 2000; Borrelli et al., 2006). Previous work by Abdulsamad et. al. has applied a variant on rSLDS (recurrent autoregressive hidden Markov models) to the optimal control of general nonlinear systems (Abdulsamad & Peters, 2023; 2020). The authors use these models to the approximate expert controllers in a closed-loop behavioural cloning context. While their algorithm focuses on value function

approximation, in contrast, we learn online without expert data and focus on flexible discrete planning.

## 3. Framework

Here, we provide a overall outline of the approach to approximate control taken with our Hybrid Hierarchical Agent (HHA) algorithm. Consider that we have decomposed the nonlinear dynamics into piecewise affine regions of the state-space using an rSLDS. Should the HHA wish to navigate to a goal specified in continuous space, the recurrent generative model parameters of the rSLDS allow it to identify the discrete region within which the goal resides, thereby lifting the goal into a high-level objective. The agent may then generate a plan at a discrete level, making use of the information-seeking bonuses that this affords. Planning translates to specifying a sequence of abstract sub-goals. Again using the recurrent generative model, the agent can specify, for each sub-goal region, a continuous point in state-space with which to drive the system into. Once in the discrete goal region, the agent straightforwardly navigates to the continuous goal. The following sections detail the components of the HHA. For additional information, please refer to Appendix. A

### 3.1. rSLDS(ro)

In the recurrent-only (ro) formulation of the rSLDS, the discrete latent states $z_t \in \{1, 2, ..., K\}$ are generated as a function of the continuous latents $x_t \in \mathbb{R}^M$ and the control input $u_t \in \mathbb{R}^N$ via a softmax regression model

$$P(z_{t+1}|x_t, u_t) = softmax(W_x x_t + W_u u_t + r) \quad (1)$$

whereby $W_x$ and $W_u$ are weight matrices with dimensions $\mathbb{R}^{K \times M}$ and $r$ is a bias of size $\mathbb{R}^K$. The continuous dynam-

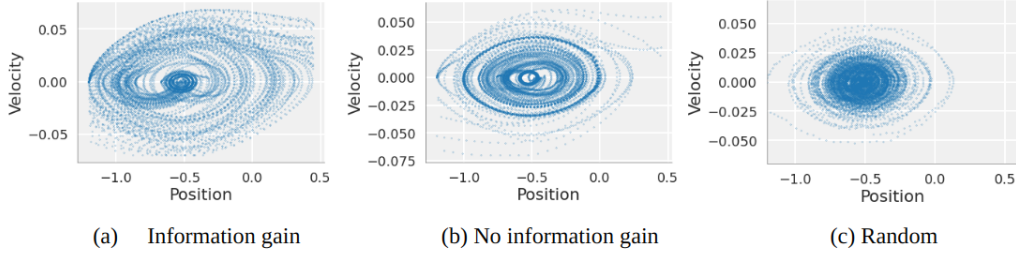(a)    Information gain    (b) No information gain    (c) Random

*Figure 2.* **HHA with information-gain explored a wider range of the state-space**. State-space coverage in Continuous Mountain Car after 10,000 steps and best of 3 runs for (a) HHA with information-gain drive, (b) HHA without information gain drive and (c) randomly sampled continuous actions baseline.

ics evolve according to a discrete linear dynamical system indexed by $z_t$ with Gaussian diagonal noise,

$$x_{t+1}|x_t, u_t, z_t = A_{z_t} x_t + B_{z_t} u_t + b_{z_t} + \nu_t,$$
$$\nu_t \sim \mathcal{N}(0, Q_{z_t}) \tag{2}$$

$$y_t|x_t = C_{z_t} x_t + \omega_t, \quad \omega_t \sim \mathcal{N}(0, S_{z_t}) \tag{3}$$

and identity emissions model with Gaussian diagonal noise. In order to learn the rSLDS parameters using Bayesian updates, conjugate matrix normal inverse Wishart (MNIW) priors are placed on the parameters of the dynamical system and recurrence weights. Inference requires approximate methods given that the recurrent connections break conjugacy rendering the conditional likelihoods non-Gaussian. Details of the Laplace Variational Expectation Maximisation algorithm used is detailed in (Zoltowski et al., 2020).

### 3.2. Discrete planner

We have a Bayesian Markov Decision Process (MDP) (Vlassis et al., 2012) described by $\mathcal{M}_B = (S, A, P_a, R, P_\theta)$. $S$ represents the set of all possible discrete states of the system and are essentially a re-description of the discrete latents $Z$ found by the rSLDS. $A$ is the set of all possible actions which, in our case, is equal to the number of states $S$. The state transition probabilities, $p_a(s_{t+1} \mid s_t = s, a_t = a, \theta) \sim Cat(\theta_{as})$, and are parameterised by $\theta \in \mathbb{R}^{s \times s \times a}$ for which we maintain Dirichlet priors over, $p(\theta_{as}) \sim Dir(\alpha_{as})$, facilitating directed exploration. Due to conjugate structure, as the agent obtains new empirical information, Bayesian updates amount to a simple count-based update of the Dirichlet parameters (Murphy, 2012). Importantly, the structure of the state transition model has been constrained by the adjacency structure of the polyhedral partitions extracted from recurrent transition dynamics of the rSLDS: invalid transitions are assigned zero probability while valid transitions are assigned a high probability (see A.5). $R$ is the reward function which, translated into the Active Inference framework, acts as a prior distribution over rewarding states

providing the agent with an optimistic bias during policy inference (Millidge et al., 2020; Parr et al., 2022).

The discrete planner outputs a discrete action, where the first action is taken from a receding horizon optimisation:

$$a_0 = \arg\min_{a_{1:T}} J(a_{1:T}) \tag{4}$$

$$J(a_{1:T}) = \mathbb{E}[\sum_{t=0}^{T} R(s_t, a_t) + IG_t(\alpha) \mid s_0, a_{1:T}]. \tag{5}$$

This includes an explicit information-seeking incentive $IG_t(\alpha)$ (see A.4). This descending discrete action $a_0$ is translated into a continuous control prior $x_j$ via the following link function,

$$x_j = \arg\max_x P(z = j|x, u) \tag{6}$$

which represents an approximately central point in the desired discrete region $j$ requested by action $a_0$ (see A.6). The ascending messages from the continuous level are translated into a categorical distribution via the rSLDS softmax link function. Importantly, the discrete planner is only triggered when the system switches into a new mode [1]. In this sense, discrete actions are temporally abstracted and decoupled from continuous clock-time in a method reminiscent of the options framework (Sutton et al., 1999; Daniel et al., 2016).

### 3.3. Continuous controller

Continuous closed-loop control is handled by a finite-horizon linear-quadratic regulator (LQR) controller. For controlling the transition from mode $i$ to mode $j$ ($x_i$ to $x_j$). The objective of the LQR controller is to minimise the following quadratic cost function:

$$\pi_{ij}(x) = \arg\min_{\pi} J_{ij}(\pi) \tag{7}$$

$$J_{ij}(\pi) = \mathbb{E}_{\pi, x_i} \Big[ (x_S - x_j)^T Q_f (x_S - x_j)$$

---

[1]Or a maximum dwell-time (hyperparameter) is reached.

$$+ \sum_{t=0}^{S-1} u_t^T R u_t \Bigg] \qquad (8)$$

where $S$ is the finite time horizon, $Q_f$ is the matrix that penalises the terminal state deviation from $x_j$ and $R$ is the control cost where high control input is penalised such that the controller only provides solutions within constraints (for further discussion, see Sec. 5). The approximate closed-loop solution to each of these sub-problems is computed offline by taking in the parameters of the linear systems indexed by the discrete modes and the continuous control priors acting as reference points (see A.3).

## 4. Results

To evaluate the performance of our (HHA) model, we applied it to the classic control problem of Continuous Mountain Car. This problem is particularly relevant for our purposes due to the sparse nature of the rewards, necessitating effective exploration strategies to achieve good performance. The HHA is initialised according to the procedure outlined in (Linderman et al., 2016). The rSLDS parameters are then fitted to the observed trajectories every 1000 steps of the environment unless a reward threshold within a single episode is reached.
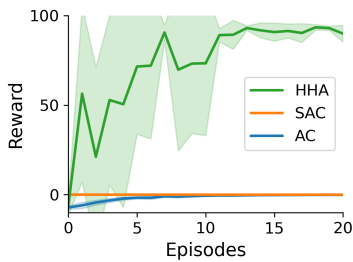


*Figure 3.* **HHA both finds the reward and captilises on its experience significantly quicker than other model-free RL baselines**. Average reward (+/- std) over 6 runs for Continuous Mountain Car (20 episodes, max episode length of 200 steps) for HHA (our model), Soft-Actor Critic (with 2 Q-functions), and Actor-Critic models. Note that after 20 episodes, SAC and AC are yet to find the reward and converge on a solution.

We find that the HHA finds piecewise affine approximations of the task-space and uses these discrete modes effectively to solve the task. Fig.1 shows that while the rSLDS has divided up the space according to position, velocity and control input, the useful modes for solving the task are those found in the position space. Once the goal and a good approximation to the system has been found, the HHA successfully and consistently navigates to the reward.

Fig. 2 shows that the HHA performs a comprehensive exploration of the state-space and significant gains in the state-space coverage are observed when using information-gain

drive in policy selection compared to without. Interestingly, even without information-gain, the area covered by the HHA is still notably better than that of the random action control. This is because the non-grid discretisation of the state-space significantly reduces the dimensionality of the search space in a behaviourally relevant way.

We compare the performance of the HHA to other reinforcement learning baselines (Actor-Critic and Soft Actor-Critic) and find that the HHA both finds the reward and captilises on its experience significantly quicker than the other models (see Fig. 3). Indeed, our model competes with the state-space coverage achieved by model-based algorithms with exploratory enhancements in the discrete Mountain Car task, which is inherently easier to solve (see A.8).

## 5. Discussion

Through the application of our Hybrid Hierarchical Agent to the Continuous Mountain Car problem, we have demonstrated that rSLDS representations hold promise for enriching planning and control. The emergence of non-grid discretisations of the state-space allows us to perform fast systems identification via enhanced exploration, and successful non-trivial planning through the delineation of abstract subgoals. Hence, the time spent exploring each region is not equivalent in euclidean space which helps mitigate the curse of dimensionality that other grid-based methods suffer from.

Such a piecewise affine approximation of the space will incur some loss of optimality in the long run when pitted against black-box approximators. This is due to the nature of caching only approximate closed-loop solutions to control within each piecewise region, whilst the discrete planner implements open-loop control. However, this approach eases the online computational burden for flexible re-planning. Hence in the presence of noise or perturbations within a region, the controller may adapt without any new computation. This is in contrast to other nonlinear model-based algorithms like model-predictive control where reacting to disturbances requires expensive trajectory optimisation at every step (Schwenzer et al., 2021). By using the piecewise affine framework, we maintain functional simplicity and interpretability through structured representation. This method is amenable to future alignment with a control-theoretic approach to safety guarantees for ensuring robust system performance and reliability.

We acknowledge there may be better solutions to dealing with control input constraints than the one given in Sec. 3.3. Different approaches have been taken to the problem of implementing constrained-LQR control, such as further piecewise approximation based on defining reachability regions for the controller (Bemporad et al., 2002).

4

## Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

## References

Abdulsamad, H. and Peters, J. Hierarchical decomposition of nonlinear dynamics and control for system identification and policy distillation. In Bayen, A. M., Jadbabaie, A., Pappas, G., Parrilo, P. A., Recht, B., Tomlin, C., and Zeilinger, M. (eds.), *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, volume 120 of *Proceedings of Machine Learning Research*, pp. 904–914. PMLR, 10–11 Jun 2020.

Abdulsamad, H. and Peters, J. Model-based reinforcement learning via stochastic hybrid models. *IEEE Open Journal of Control Systems*, 2:155–170, 2023. doi: 10.1109/OJCSYS.2023.3277308.

Bemporad, A., Borrelli, F., and Morari, M. Piecewise linear optimal controllers for hybrid systems. In *Proceedings of the 2000 American Control Conference. ACC (IEEE Cat. No.00CH36334)*, volume 2, pp. 1190–1194 vol.2, 2000. doi: 10.1109/ACC.2000.876688.

Bemporad, A., Morari, M., Dua, V., and Pistikopoulos, E. N. The explicit linear quadratic regulator for constrained systems. *Automatica*, 38(1):3–20, 2002. ISSN 0005-1098. doi: https://doi.org/10.1016/S0005-1098(01)00174-1.

Borrelli, F., Bemporad, A., Fodor, M., and Hrovat, D. An mpc/hybrid system approach to traction control. *IEEE Transactions on Control Systems Technology*, 14(3):541–552, 2006. doi: 10.1109/TCST.2005.860527.

Coulom, R. Efficient selectivity and backup operators in monte-carlo tree search. In van den Herik, H. J., Ciancarini, P., and Donkers, H. H. L. M. J. (eds.), *Computers and Games*, pp. 72–83, Berlin, Heidelberg, 2007. Springer Berlin Heidelberg. ISBN 978-3-540-75538-8.

Daniel, C., van Hoof, H., Peters, J., and Neumann, G. Probabilistic inference for determining options in reinforcement learning. *Machine Learning*, 104(2):337–357, September 2016. ISSN 1573-0565. doi: 10.1007/s10994-016-5580-x.

d'Avila Garcez, A. and Lamb, L. C. Neurosymbolic ai: the 3rd wave. *Artificial Intelligence Review*, 56(11):12387–12406, November 2023. ISSN 1573-7462. doi: 10.1007/s10462-023-10448-w.

Friston, K., Da Costa, L., Tschantz, A., Kiefer, A., Salvatori, T., Neacsu, V., Koudahl, M., Heins, R., Sajid,

N., Markovic, D., Parr, T., Verbelen, T., and Buckley, C. Supervised structure learning. 12 2023.

Gobet, F., Lane, P., Croker, S., Cheng, P., Jones, G., Oliver, I., and Pine, J. Chunking mechanisms in human learning. *Trends in cognitive sciences*, 5:236–243, 07 2001. doi: 10.1016/S1364-6613(00)01662-4.

Gou, S. Z. and Liu, Y. DQN with model-based exploration: efficient learning on environments with sparse rewards. *CoRR*, abs/1903.09295, 2019.

Heins, C., Millidge, B., Demekas, D., Klein, B., Friston, K., Couzin, I., and Tschantz, A. pymdp: A python library for active inference in discrete state spaces. *arXiv preprint arXiv:2201.03904*, 2022.

LaValle, S. M. *Planning Algorithms*, chapter 2. Cambridge University Press, Cambridge, 2006.

Liberzon, D. *Switching in Systems and Control*. Systems & Control: Foundations & Applications. Birkhäuser Boston, 2003. ISBN 9780817642976.

Linderman, S. W., Miller, A. C., Adams, R. P., Blei, D. M., Paninski, L., and Johnson, M. J. Recurrent switching linear dynamical systems, October 2016.

Millidge, B., Tschantz, A., Seth, A. K., and Buckley, C. L. On the relationship between active inference and control as inference, 2020.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A. K., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. Human-level control through deep reinforcement learning. *Nature*, 518:529–533, 2015.

Murphy, K. P. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Newell, A. and Simon, H. A. *Human Problem Solving*. Prentice-Hall, Englewood Cliffs, NJ, 1972.

OpenAI. Continuous mountain car environment, 2021. Accessed: 2024-05-25.

Parr, T., Pezzulo, G., and Friston, K. *Active Inference: The Free Energy Principle in Mind, Brain, and Behavior*. MIT Press, 2022. ISBN 9780262045353.

Schwenzer, M., Ay, M., Bergs, T., and Abel, D. Review on model predictive control: an engineering perspective. *The International Journal of Advanced Manufacturing Technology*, 117(5):1327–1349, November 2021. ISSN 1433-3015. doi: 10.1007/s00170-021-07682-3.

Sutton, R. S., Precup, D., and Singh, S. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial Intelligence*, 112(1): 181–211, 1999. ISSN 0004-3702. doi: https://doi.org/10.1016/S0004-3702(99)00052-1.

Vlassis, N., Ghavamzadeh, M., Mannor, S., and Poupart, P. *Bayesian Reinforcement Learning*, pp. 359–386. Springer Berlin Heidelberg, Berlin, Heidelberg, 2012. ISBN 978-3-642-27645-3. doi: 10.1007/978-3-642-27645-3_11.

Zoltowski, D. M., Pillow, J. W., and Linderman, S. W. Unifying and generalizing models of neural dynamics during decision-making, 2020.

# A. Appendix / supplemental material

### A.1. Framework

### Optimal Control

We adopt the optimal control framework, specifically we consider discrete time state space dynamics of the form:

$$x_{t+1} = f(x_t, u_t, \eta_t) \tag{9}$$

with known initial condition $x_0$, and $\eta_t$ drawn from some time invariant distribution $\eta_t \sim D$, where $f$ we assume $p(x_{t+1} \mid x_t, u_t)$ is a valid probability density throughout.

We use $c_t : X \times U \to \mathbb{R}$ for the control cost function at time $t$ and let $\mathbb{U}$ be the set of admissible (non-anticipative, continuous) feedback control laws, possibly restricted by affine constraints. The optimal control law for the finite horizon problem is given as:

$$J(\pi) = \mathbb{E}_{x_0, \pi}\left[\sum_{t=0}^{T} c_t(x_t, u_t)\right] \tag{10}$$

$$\pi^* = \arg\min_{\pi \in \mathbb{U}} J(\pi) \tag{11}$$

### PWA Optimal Control

The fact we do not have access to the true dynamical system $f$ motivates the use of a piecewise affine (PWA) approximation. Also known as hybrid systems:

$$x_{t+1} = A_i x_t + B_i u_t + \epsilon_t \tag{12}$$

$$\text{when } (x_t, u_t) \in H_i \tag{13}$$

Where $\mathbb{H} = \{H_i : i \in [K]\}$ is a polyhedral partition of the space $X \times U$.

In the case of a quadratic cost function, it can be shown the optimal control law for such a system is peicewise linear. Further there exist many completeness (universal approximation) type theorems for peicewise linear approximations implying if the original system is controllable, there will exist a peicewise affine approximation through which the system is still controllable (Bemporad et al., 2000; Borrelli et al., 2006).

### Relationship to rSLDS(ro)

We perform a canonical decomposition of the control objective $J$ in terms of the components or modes of the system. By slight abuse of notation $[x_t = i] := [(x_t, u_t) \in H_i]$ represent the Iverson bracket.

$$J(\pi) = \sum_t \int p_\pi(x_t \mid x_{t-1}, u_t) c_t(x_t, u_t) dx_t dx_{t-1} \tag{14}$$

$$= \sum_t \int \sum_{i \in [K]} [x_{t-1} = i] p_\pi(x_t \mid x_{t-1}, u_t) c_t(x_t, u_t) dx_t dx_{t-1} \tag{15}$$

$$\tag{16}$$

Now let $z_t$ be the random variable on $[K]$ induced by $Z_t = i$ if $[x_t = i]$ we can rewrite the above more concisely as,

$$J(\pi) = \sum_t \int \sum_{i \in [K]} p_\pi(x_t, z_{t-1} = i \mid x_{t-1}, u_t) c_t(x_t, u_t) dx_t dx_{t-1} \tag{17}$$

$$= \sum_{i \in [K]} \sum_t \int p_\pi(x_t, z_{t-1} = i \mid x_{t-1}, u_t) c_t(x_t, u_t) dx_t dx_{t-1} \tag{18}$$

$$= \sum_{i \in [K]} \sum_t \mathbb{E}_{\pi_i}[c_t(x_t, u_t)] \tag{19}$$

$$(20)$$

which is just the expectation under a recurrent dynamical system with deterministic switches. Later (see A.4), we exploit the non-deterministic switches of rSLDS in order to drive exploration.

### A.2. Hierarchical Decomposition

Our aim was to decouple the discrete planning problem from the fast low-level controller. In order to break down the control objective in this manner, we first create a new discrete variable which simply tracks the transitions of $z$, this allows the discrete planner to function in a temporally abstracted manner.

**Decoupling from clock time** Let the random variable $(\zeta_s)_{s>0}$ record the transitions of $(z_t)_{t>0}$ i.e. let

$$\tau_s(\tau_{s-1}) = \min\{t : z_{t+1} \neq z_t, t > \tau_{s-1}\}, \tau_0 = 0 \tag{21}$$

be the sequence of first exit times, then $\zeta$ is given by $\zeta_s = z_{\tau_s}$.

With these variables in hand, we frame a small section of the global problem as a first exit problem.

**Low level problem** Consider the first exit problem defined by,

$$\pi_{ij}(x_0) = \arg\min_{\pi, S} J_{ij}(\pi, x_0, S) \tag{22}$$

$$J_{ij}(\pi, x_0, S) = \mathbb{E}_{\pi, x_0}[\sum_{t=0}^{S} c(x_t, u_t)] \tag{23}$$

$$\text{s.t. } (x_t, u_t) \in H_i \tag{24}$$

$$\text{s.t. } c(x, u) = 0 \text{ when } (x, u) \in \partial H_{ij} \tag{25}$$

where $\partial H_{ij}$ is the boundary $H_i \bigcap H_j$.

Due, to convexity of the polyhedral partition, the full objective admits the decomposition into subproblems

$$J(\pi) = \sum_s J_{\zeta(s+1), \zeta(s)}(\pi) \tag{26}$$

$$\tag{27}$$

**Slow and fast modes** The goal is to tackle the decomposed objectives individually, however the hidden constraint that the trajectories line up presents a computational challenge. Here we make the assumption that the difference in cost induced by different starting positions, induces a relatively small change in the minimum cost $J_{ij}$, intuitively this happens if the minimum state cost in each mode is relatively uniform as compared to the difference between regions.

**High level problem** If the above assumption holds, we let $J_{ij}^* = \min_\pi \int_{x_0} J_{ij}(\pi, x_0)p(x_0)$ be the average cost of each low-level problem. We form a markov chain:

$$p_{ik}(u) = \mathbb{P}(\zeta_{s+1} = k \mid \zeta_s = i, \pi_{ij}^*, u^d = j) \tag{28}$$

and let $p_{\pi_d}$ be the associated distribution over trajectories induced by some discrete state feedback policy, along with the discrete state action cost $c_d(u^d = j, \eta = i) = J_{ij}^*$ we may write the high level problem:

$$\pi_d^* = \min_{\pi_d} J_d(\pi, \eta_0) \tag{29}$$

$$= \mathbb{E}p_{\pi_d}[\sum_{s=0}^{S} c_d(\eta_s, u_s^d)] \tag{30}$$

Our approximate control law is then given by $\pi_{ij}^* \circ \pi_d^* \circ id(x)$

## A.3. Offline Low Level Problems: Linear Quadratic Regulator (LQR)

Rather than solve the first-exit problem directly, we formulate an approximate problem by finding trajectories that end at specific 'control priors' (see A.6). Recall the low level problem given by:

$$\pi_{ij}(x_0) = \arg\min_{\pi, S} J_{ij}(\pi, x_0, S) \tag{31}$$

$$J_{ij}(\pi, x_0, S) = \mathbb{E}_{\pi, x_0}[\sum_{t=0}^{S} c(x_t, u_t)] \tag{32}$$

$$\text{s.t. } (x_t, u_t) \in H_i \tag{33}$$

$$\text{s.t. } c(x, u) = 0 \text{ when } (x, u) \in \partial H_{ij} \tag{34}$$

In order to approximate this problem with one solvable by a finite horizon LQR controller, we adopt a fixed goal state, $x^* \in H_j$. Imposing costs $c_t(x_t, u_t) = u_t^T R u_t$ and $c_S(x_S, u_S) = (x - x^*)Q_f(x - x^*)$. Formally we solve,

$$\pi_{ij}(x_0) = \arg\min_{\pi, S} J_{ij}(\pi, x_0, S) \tag{35}$$

$$J_{ij}(\pi, x_0, S) = \mathbb{E}_{\pi, x_0}[(x_S - x^*)^T Q_f (x_S - x^*) + \sum_{t=0}^{S-1} u_t^T R u_t] \tag{36}$$

$$\tag{37}$$

by integrating the discrete Ricatti equation backwards. Numerically, we found optimising over different time horizons made little difference to the solution, so we opted to instead specify a fixed horizon (hyperparameter). These solutions are recomputed offline every time the linear system matrices change.

**Designing the cost matrices** Instead of imposing the state constraints explicitly, we record a high cost which informs the discrete controller to avoid them. In order to approximate the constrained input we choose a suitably large control cost $R = rI$. We adopted this approach for the sake of simplicity, potentially accepting a good deal of sub-optimality. However, we believe more involved methods for solving input constrained LQR could be used in future, e.g. (Bemporad et al., 2000), especially because we compute these solutions offline.

## A.4. Online high level problem

The high level problem is a discrete MDP with a 'known' model, so the usual RL techniques (approximate dynamic programming, policy iteration) apply. Here, however we choose to use a model-based algorithm with a receding horizon inspired by Active Inference, allowing us to easily incorporate exploration bonuses.

Let the Bayesian MDP be given by $\mathcal{M}_B = (S, A, P_a, R, P_\theta)$ be the MDP, where $p_a(s_{t+1} \mid s_t, a_t, \theta) \sim Cat(\theta_{as})$ and $p(\theta_{as}) \sim Dir(\alpha)$ We estimate the open loop reward plus optimistic information theoretic exploration bonuses

**Active Inference conversion** We adopt the Active Inference framework for dealing with exploration. Accordingly we adopt the notation $\ln \tilde{p}(s_t, a_t) = R(s_t, a_t)$ and refer to this 'distribution' as the goal prior (Millidge et al., 2020), and optimise over open loop policies $\pi = (a_0, ..., a_T)$.

$$J(a_{1:T}, s_0) = \mathbb{E}[\sum_{t=0}^{T} R(s_t, a_t) + IG_p + IG_s \mid s_0, a_{1:T}] \tag{38}$$

where parameter information-gain is given by $IG_p = D_{KL}[p_{t+1}(\theta) \mid\mid p_t(\theta)]$, with $p_t(\theta) = p(\theta \mid s_{0:t})$. In other words, we add a bonus when we expect the posterior to diverge from the prior, which is exactly the transitions we have observed least (Heins et al., 2022).

We also have a state information-gain term, $IG_s = D_{KL}[p_{t+1}(s_{t+1}) \mid\mid p_t(s_{t+1})]$. In this case (fully observed), $p_{t+1}(s_{t+1}) = \delta_s$ is a one-hot vector. Leaving the term $\mathbb{E}_t[-\ln p_t(s_{t+1})]$ leading to a maximum entropy term (Heins et al., 2022).

We calculate the above with Monte Carlo sampling which is possible due to the relatively small number of modes. Local approximations such as Monte Carlo Tree Search could easily be integrated in order to scale up to more realistic problems.

Alternatively, for relatively stationary environments we could instead adopt approximate dynamic programming methods for more habitual actions.

### A.5. Extracting the adjacency matrix from rSLDS

In order to generate the possible transitions from the rSLDS, we calculate the set of active constraints for each region from the softmax representation, $p(z \mid x) = \sigma(Wx + b)$. Specifically to check region $i$ is adjacent to region $j$ we verify the solution linear program:

$$-b_j = \min(W_i - W_j)x \tag{39}$$
$$\text{s.t. } (W_i - W_k)x \leq (b_i - b_k) \ \forall k \in [K] \tag{40}$$
$$\text{s.t. } x \in (x_{lb}, x_{ub}) \tag{41}$$

Where $(x_{lb}, x_{ub})$ are bounds chosen to reflect realistic values for the problem. This ensures we only lift transitions to the discrete model, if they are possible. Again, these can be calculated offline.

We initialise the entries of the transition model in the discrete MDP for possible transitions to 0.9 facilitating guided-exploration via information-gain through a count-based updates to the transition priors.

### A.6. Generating continuous control priors

In order to generate control priors for the LQR controller which correspond to each of the discrete states we must find a continuous state $x_i$ which maximises the probability of being in a desired $z$:

$$x_i = \arg\max_x P(z = i|x, u) \tag{42}$$

For this we perform a numerical optimisation in order to maximise this probability. Consider that this probability distribution $P(z = i|x)$ is a softmax function for the i-th class is defined as:

$$\sigma(v_i) = \frac{\exp(v_i)}{\sum_j \exp(v_j)}, v_i = w_i \cdot x + r_i \tag{43}$$

where $w_i$ is the i-th row of the weight matrix, $x$ is the input and $r_i$ is the i-th bias term. The update function used in the gradient descent optimisation can be described as follows:

$$x \leftarrow x + \eta \nabla_x \sigma(v_i) \tag{44}$$

where $\eta$ is the learning rate and the gradient of the softmax function with respect to the input vector $x$ is given by:

$$\nabla_x \sigma(v_i) = \frac{\partial \sigma(v_i)}{\partial v} \cdot \frac{\partial v}{\partial x} = \sigma(v_i)(\mathbf{e}_i - \sigma(v)) \cdot W \tag{45}$$

in which $\sigma(v)$ is the vector of softmax probabilities, and $\mathbf{e}_i$ is the standard basis vector with 1 in the i-th position and 0 elsewhere. The gradient descent process continues until the probability $P(z = i|x)$ exceeds a specified threshold $\theta$ which we set to be 0.7. This threshold enforces a stopping criterion which is required for the cases in which the region $z$ is unbounded.

## A.7. Model-free RL baselines

### A.7.1. SOFT-ACTOR CRITIC WITH 2 Q-FUNCTIONS

*Table 1.* Summary of the Soft Actor-Critic algorithm with multiple Q-functions.

| COMPONENT | INPUT |
|---|---|
| Q-NETWORK | 3×256×256×256×2 |
| POLICY NETWORK | 2×256×256×256×2 |
| ENTROPY REGULARIZATION COEFF | 0.2 |
| LEARNING RATES (QNET + POLNET) | 3E-4 |
| BATCHSIZE | 60 |

### A.7.2. ACTOR-CRITIC

*Table 2.* Summary of the Actor-Critic algorithm

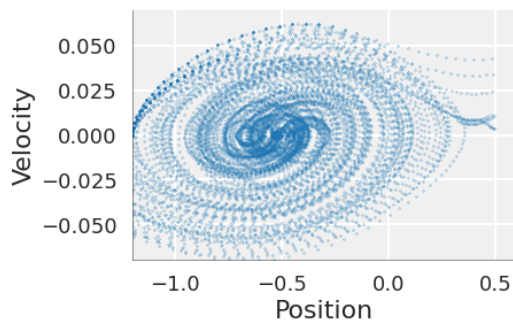| COMPONENT | INPUT |
|---|---|
| FEATURE PROCESSING | STANDARDSCALER, RBF KERNELS (4 × 100) |
| VALUE-NETWORK | 4001 PARAMETERS (1 DENSE LAYER) |
| POLICY NETWORK | 802 PARAMETERS (2 DENSE LAYERS) |
| GAMMA | 0.95 |
| LAMBDA | 1E-5 |
| LEARNING RATES (POLICY + VALUE) | 0.01 |

## A.8. Model-based RL baseline

### A.8.1. A DEEP Q-NETWORK WITH MODEL-BASED EXPLORATION (DQN-MBE)
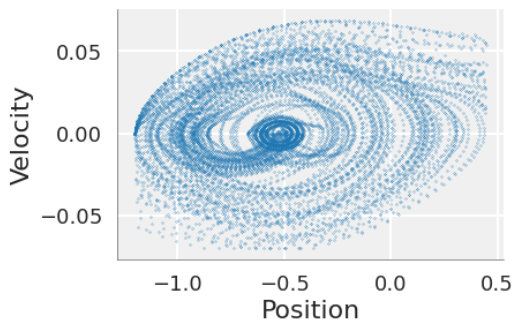


(a) HHA (our model) on Continuous Mountain Car
(b) DQN-MBE on Discrete Mountain Car

*Figure 4.* **On Continuous Mountain Car, our model (HHA) competes with the state-space coverage achieved by model-based baselines on Discrete Mountain Car (an easier problem)** State-space coverage after 10,000 timesteps on (a) Continuous Mountain Car task using our model (HHA) and (b) Discrete Mountain Car task using a Deep Q-Network with Model-Based Exploration (DQN-MBE) (Gou & Liu, 2019). Exact parameters in Table 3.

*Table 3.* Summary of DQN-MBE algorithm (Gou & Liu, 2019)

| COMPONENT | INPUT |
|---|---|
| Q-NETWORK | 1 HIDDEN-LAYER, 48 UNITS, ReLU |
| DYNAMICS PREDICTOR NETWORK (FULLY CONNECTED) | 2 HIDDEN-LAYERS (EACH 24 UNITS), ReLU |
| $\epsilon$ MINIMUM | 0.01 |
| $\epsilon$ DECAY | 0.9995 |
| REWARD DISCOUNT | 0.99 |
| LEARNING RATES (QNET / DYNAMICS-NET) | 0.05 / 0.02 |
| TARGET Q-NETWORK UPDATE INTERVAL | 8 |
| INITIAL EXPLORATION ONLY STEPS | 10000 |
| MINIBATCH SIZE (Q-NETWORK) | 16 |
| MINIBATCH SIZE (DYNAMICS PREDICTOR NETWORK) | 64 |
| NUMBER OF RECENT STATES TO FIT PROBABILITY MODEL | 50 |