Adaptive World Models: Learning Behaviors by Latent Imagination Under Non-Stationarity

Emiliyan Gospodinov[1*] **Vaisakh Shaj**[1] **Philipp Becker**[1] **Stefan Geyer**[2]
Gerhard Neumann[1]
[1]Karlsruhe Institute of Technology (KIT), Germany
[2]Institute for Artificial Intelligence, Stuttgart, Germany

Abstract

Developing foundational world models is a key research direction for embodied intelligence, with the ability to adapt to non-stationary environments being a crucial criterion. In this work, we introduce a new formalism, Hidden Parameter-POMDP, designed for control with adaptive world models. We demonstrate that this approach enables learning robust behaviors across a variety of non-stationary RL benchmarks. Additionally, this formalism effectively learns task abstractions in an unsupervised manner, resulting in structured, task-aware latent spaces.

1 Introduction

Recent advances in foundational models have achieved remarkable success in NLP and vision tasks [18, 19]. Still, they fall short in addressing the complexities faced by embodied agents in dynamic, real-world environments. For embodied intelligence, we argue that it is essential to develop foundational world models [7, 21] that capture the causal nature of the world we live in and can make counterfactual predictions. Furthermore, these models should adapt dynamically to non-stationary environments.

Current state-of-the-art approaches for Model-Based Reinforcement Learning (MBRL) [8, 9, 13] often use probabilistic state-space models [17, 2, 22] as a backbone. They learn behaviors by making counterfactual predictions in the latent space of world models. Often these approaches focus on agents mastering a specific, narrow task. Throughout this work, a "**task**" refers to a particular schema of environment dynamics or a specific reward function. In real-world settings, however, tasks are frequently non-stationary and subject to change over time. Thus, a truly intelligent agent must (1) understand the current task and (2) dynamically adapt its perception, model, and behavior to new tasks with minimal interaction.

We identified a gap in the literature regarding MBRL in latent spaces that address multitask learning and adaptation under non-stationarity. This work makes two key contributions: (1) highlighting the limitations of current state-of-the-art model-based agents in non-stationary settings, and (2) proposing a new formalism that models non-stationarity as an additional causal latent variable, resulting in robust policies.

[*]Corresponding author. Email to <gospodinov.emilian@gmail.com>

2 Non-Stationary RL Formalisms In Latent Spaces

2.1 POMDP formalism

Existing state-of-the-art MBRL agents [8, 9, 11, 13, 15] that learn in latent spaces typically rely on the partially observable Markov decision process (POMDP) formalism. In this framework, incoming sensory signals are used to update the agent's belief about the hidden state of the environment, enabling the agent to make decisions under uncertainty. Theoretically, the POMDP formalism could handle non-stationarity by treating slowly changing, unobserved tasks as part of the latent states [31]. Here the assumption is that the underlying environment is assumed to be stationary, but the agent has an incomplete view of it [16]. Consequently, single-task frameworks that rely on latent dynamics models for learning should, in theory, be applicable in streaming settings. However, an unstructured latent state, without inductive biases, may hinder the learning of sample-efficient adaptive policies.

2.2 HiP-POMDP formalism

A complementary but more popular view in literature for non-stationary RL is that the components of the RL (transition, reward, observation functions, action space, etc.) may depend upon time [16]. We build our formalism, the HiP-POMDP, upon this non-stationary function view [5, 32, 31, 23, 4, 24]. We start by providing a formal definition of a HiP-POMDP and demonstrate that this simple modification, along with a scalable variational inference scheme, enables learning adaptive policies across a wide range of non-stationary scenarios where the changing tasks are unknown to the agent.

Definition 2.1. *A HiP-POMDP is given by a tuple*

$$\{\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{C}, \mathcal{L}, p_s(\boldsymbol{s}_{t+1}|\boldsymbol{a}_t, \boldsymbol{s}_t, \boldsymbol{l}), p_o(\boldsymbol{o}_t|\boldsymbol{s}_t, \boldsymbol{l}), r(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{l}), p_c(\mathcal{C}_l|\boldsymbol{l})\},$$

where $\mathcal{S}$, $\mathcal{A}$, and $\mathcal{O}$ are the standard state, action and observation spaces. Additionally, we introduce a space of latent task variables $\mathcal{L}$, where $\boldsymbol{l} \in \mathcal{L}$, and a space of context observations $\mathcal{C}$, where $\mathcal{C}_l \in \mathcal{C}$. Context observations $\mathcal{C}_l$ are generated from $\boldsymbol{l}$ according to $p(\mathcal{C}_l|\boldsymbol{l})$. Finally, the transition model p_s, observation model p_o, and the reward function r all depend on the latent task l.

This general definition does not specify how exactly the context can be observed. Throughout this work, we assume $\mathcal{C}_l = \{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o}')_n\}_{n=1}^N$, i.e., a set of N recent transitions. However, more expressive $\mathcal{C}_l$ including temporal embeddings, task metadata, or any other available information about the task could be used in the future. In a HiP-POMDP, the agent's objective is to infer a latent task distribution $p(\boldsymbol{l} \mid \boldsymbol{C}_l)$ based on the context observations $\boldsymbol{C}_l$ and learn a latent task conditional policy $\pi(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{l})$ that maximizes the expected cumulative discounted reward, $\mathbb{E}_\pi \left[\sum_{t=0}^{T-1} \gamma^t r_{t+1}\right]$, where T is the total number of time steps and $\gamma \in [0, 1]$ is the discount factor.

3 Adaptive Latent Space Models for HiP-POMDPs

In line with standard practices in model-based reinforcement learning (MBRL), we alternate between representation learning, behavior learning, and environment interactions to learn policies in the latent space of a world model. However, unlike existing approaches, we make each of these stages adaptive by conditioning them on an inferred task representation or abstraction. Thus, our work goes in the direction of building foundational multi-task world models and subsequent behavior policies. For efficient learning and inference, we adopt a two-phase approach, where we first infer the latent task, which we then use to condition the model, actor, and critic.

3.1 Inferring Latent Task Abstractions via Aggregation

As stated, throughout this work, we choose $\boldsymbol{C_l}$ to be a collection of N recently observed transition tuples $\{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o}')_n\}_{n=1}^N$, practically implemented as a FIFO buffer. Note that

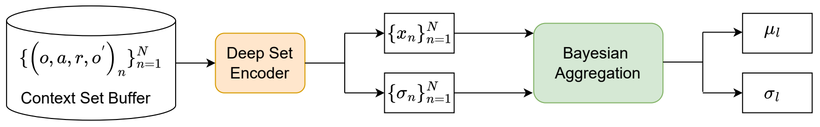


Figure 1: Given a set of N transitions, the deep set encoder emits a latent representation for each of the observations and their corresponding uncertainty. The set of latent representations is then aggregated via Bayesian aggregation to infer $p(\boldsymbol{l} \mid \boldsymbol{C_l})$.

we chose these tuples of observations, actions, rewards, and next observations because they worked well in practice, capturing sufficient task-relevant statistics as shown in Section 4.

Now, to form the posterior belief over the latent task variable $\boldsymbol{l}$, we first extract encoded representations $\boldsymbol{x}_n$ with associated variances $\boldsymbol{\sigma}_n$ from each transition tuple in the context set using a set encoder network with shared parameters. We assume the latent representation is distributed according to $\mathcal{N}(\boldsymbol{x}_n \mid \boldsymbol{l}, \operatorname{diag}(\boldsymbol{\sigma}_l))$. This assumption allows us to form Gaussian beliefs $\mathcal{N}(\boldsymbol{\mu}_l, \boldsymbol{\sigma}_l)$ over $\boldsymbol{l}$ using Bayes rule. As shown in [30, 23], the beliefs over $\boldsymbol{l}$ can be computed in closed form, given a Gaussian prior $p_0(\mathbf{l})$. The update rules and their properties are detailed in Appendix B.

3.2 Learning Adaptive Representations

In this stage, we learn representations of generative world models that can make counterfactual predictions of the world states based on imagined actions. We make these learned representations adaptive to the task at hand based on the generative model shown in Figure 2. We achieve this by maximizing the conditional data log-likelihood and subsequently deriving an evidence lower bound, as in Equation 1. A detailed derivation can be found in Appendix A.

$$\ln p(\boldsymbol{o}_{1:T}, r_{1:T} \mid \boldsymbol{a}_{1:T}, \boldsymbol{C}_l) \geq \sum_{t=1}^{T} \underbrace{\mathbb{E}_{p(\boldsymbol{l}|\boldsymbol{C}_l)q(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l})}\left[\ln p(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l})\right]}_{\textbf{Reconstruction Term}}$$
$$+ \underbrace{\mathbb{E}_{p(\boldsymbol{l}|\boldsymbol{C}_l)q(\boldsymbol{s}_{t-1}|\boldsymbol{o}_{\leq t-1},\boldsymbol{a}_{<t-1},\boldsymbol{l})}\left[\mathrm{D_{KL}}\left(q(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}) \,\|\, p(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l})\right)\right]}_{\textbf{Regularization Term}} \quad (1)$$

The outer expectation can be estimated using a reparameterized sample from the latent task posterior $p(\boldsymbol{l} \mid \boldsymbol{C}_l)$. The practical implementation of this builds upon the RSSMs used in the popular Dreamer series of models [8, 9, 11, 12], where an additional deterministic path (using a GRU) is used in addition to the stochastic SSM for long-term predictions. The subsequent Hidden Parameter-RSSM generative model is shown in Figure 2.

Discussion: Though we use the generative model from [8, 9], the HiP-POMDP formalism can be used in conjunction with any model-based RL framework in latent spaces [1, 11–14, 20] for multitask learning.

3.3 Learning Adaptive Behaviors

The agent optimizes long-term rewards using a context-sensitive actor-critic approach [25, 8], conditioning both actor and critic on the latent task representation $\boldsymbol{l}$. The actor $\pi_{\boldsymbol{\phi}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{l})$ selects actions to maximize expected values along imagined trajectories, while the critic $v_{\boldsymbol{\psi}}(\boldsymbol{s}_\tau, \boldsymbol{l})$ regresses

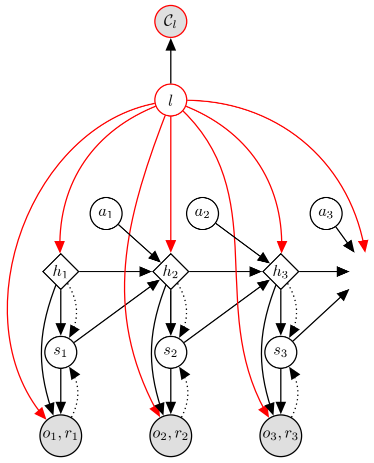


Figure 2: Hidden Parameter RSSM: The latent task variable is inferred from context $\boldsymbol{C_l}$ via Bayesian aggregation. Solid lines indicate the generative process and dashed lines the inference model. Modifications from [8] are shown in red.

those estimates:

$$\max_{\boldsymbol{\phi}} \ \mathbb{E}_{q_{\boldsymbol{\theta}},\pi_{\boldsymbol{\phi}}}\left(\sum_{\tau=t}^{t+H} \mathrm{V}_\lambda(\boldsymbol{s}_\tau, \boldsymbol{l})\right), \quad \min_{\boldsymbol{\psi}} \ \mathbb{E}_{q_{\boldsymbol{\theta}},\pi_{\boldsymbol{\phi}}}\left(\sum_{\tau=t}^{t+H} \frac{1}{2}\left\| v_{\boldsymbol{\psi}}(\boldsymbol{s}_\tau, \boldsymbol{l}) - \mathrm{V}_\lambda(\boldsymbol{s}_\tau, \boldsymbol{l})\right\|^2\right).$$

Here, $\mathrm{V}_\lambda(\boldsymbol{s}_\tau, \boldsymbol{l})$ is the λ-return [25, 8], a smoothed estimate of the cumulative reward that balances short- and long-term returns using the discount factor λ. We compute analytic gradients through the learned dynamics to optimize the actor via stochastic backpropagation through time.

Discussion As shown in Figure 2, during a short imagination rollout the latent task $\boldsymbol{l}$ remains fixed. This is a reasonable assumption for such short horizons. However, during environment interaction, the context buffer is updated continuously. This enables the agent to re-infer the task and adapt to both inter and intra-episodic task changes.

4 Evaluation

In this section, we evaluate the performance of two competing formalisms—POMDP and HiP-POMDP—in handling non-stationarity within an episodic evaluation setting. We focus on three broad categories of non-stationarity: (1) changing transition functions, (2) changing rewards, and (3) a combination of both. For each category, we further consider two scenarios:

- **Inter-Episodic Non-Stationarity:** Changes remain fixed within an episode but vary between episodes.
- **Intra-Episodic Non-Stationarity:** Non-stationary changes can occur within a single episode.

A more detailed description of these scenarios is provided in Appendix C. In all experiments, proprioceptive sensors are used as the source of observations.

Algorithms Compared: We use the Dreamer [8] as our baseline for the POMDP formalism. For the HiP-POMDP, we modify Dreamer by incorporating latent task abstractions, ensuring a fair comparison between the two approaches. Additionally, we include an "Oracle" baseline where the task is assumed to be directly observed. In this setup, the known task replaces the inferred latent task variable $\boldsymbol{l}$, serving as an upper bound on performance. This helps illustrate the potential gains if perfect task information were available.

We evaluate the agents in all experiments by calculating the mean return from 10 trajectories every 25 epochs, each with randomly sampled environmental changes. The performance curves are computed by averaging the results over 10 different random seeds. Our evaluation answers the following questions:

Can HiP-POMDP agents handle changing dynamics? To evaluate the effectiveness of HiP-POMDP formalism under changing dynamics, here we introduce two tasks: 1) We modify the standard HalfCheetah agent by adding joint perturbations of varying magnitudes randomly, and 2) the Hopper agent by randomly changing the body mass and inertia for random number of body parts. Additional evaluation is introduced in Appendix C.3.

As seen in Figure 3 HiP-POMDP agent results in robust performance gains, especially under challenging intra-episodic changes and even competing with the Oracle.

Can HiP-POMDP agents handle changing objectives? To create non-stationarity with changing reward functions/objectives, we modify the standard HalfCheetah such that a target velocity needs to be reached which changes randomly. Additionally, we evaluate the agents on custom-designed multi-task benchmarks using pre-defined tasks from [15], where each task requires the agent to perform different skills (e.g., standing, running, flip) in various directions. As such multi-skill objective changes are more challenging, the experiments are run over 5M steps.

As seen in Figure 4 the vanilla POMDP agent fails to deal with objective changes in all cases. On the other hand side, the HiP-POMDP agent with inferred task abstractions resolves the

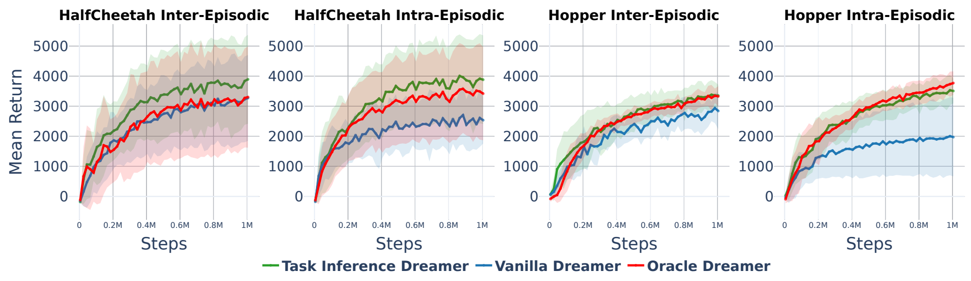


Figure 3: Performance of HalfCheetah and Hopper agents under changing dynamics caused by joint perturbations and body mass inertia variations, respectively.

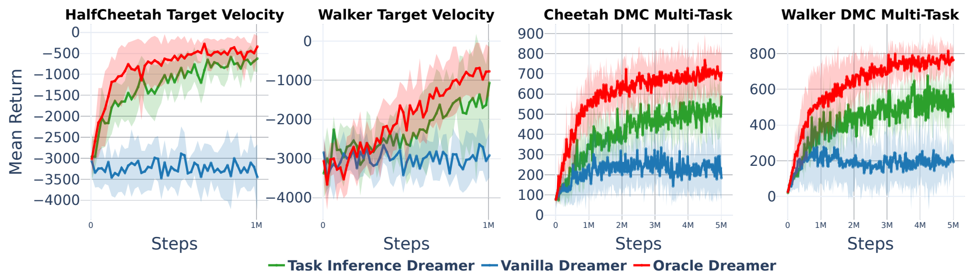


Figure 4: Performance comparison of Half Cheetah and Walker agents under different changing reward scenarios (changing target velocities and skills).

issue to a large extent. Further investigation as well as evaluation under combined changes can be found in Appendix C.4 and C.5 respectively.

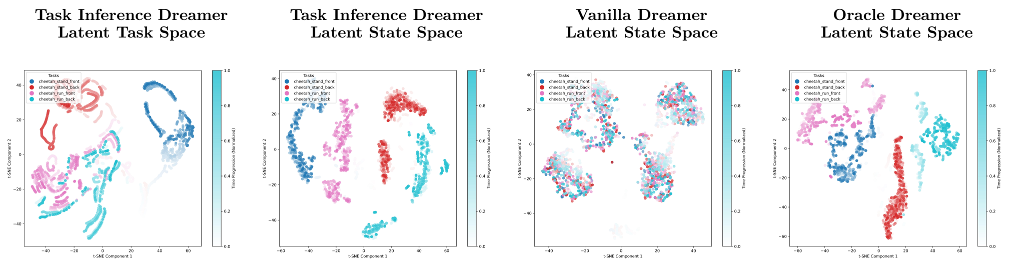


Figure 5: 2d projections of learned latent state spaces on DMC Cheetah learning 4 skills, Table 1.

Does the HiP-POMDP agents learn meaningful latent representations? Finally, we compared the learned state-space representations (the concatenation of $\boldsymbol{s_t}$ and $\boldsymbol{h_t}$) of the world model (RSSM) in both POMDP and HiP-POMDP settings. As shown in Figure 5, the task abstractions in HiP-POMDP shape a more structured and disentangled latent space that aligns with the inferred tasks, unlike the POMDP setting. This disentanglement was also observed in the latent task space representation ($\boldsymbol{l}$) within the HiP-POMDP setup.

5 Conclusion

In this work, we introduced the HiP-POMDP formalism to learn adaptive world models and behavior policies in latent state spaces. The formalism resulted in algorithms that learn meaningful task abstractions and improved performance on a variety of non-stationary benchmarks. Future work would extend these models to more high-dimensional sensory inputs like images and point clouds.

Acknowledgments

The authors acknowledge support by the state of Baden-Württemberg through bwHPC, as well as the HoreKa supercomputer funded by the Ministry of Science, Research and the Arts Baden-Württemberg and by the German Federal Ministry of Education and Research.

References

[1] P. Becker and G. Neumann. On uncertainty in deep state space models for model-based reinforcement learning. *arXiv preprint arXiv:2210.09256*, 2022.

[2] P. Becker, H. Pandya, G. Gebhardt, C. Zhao, C. J. Taylor, and G. Neumann. Recurrent kalman networks: Factorized inference in high-dimensional deep feature spaces. In *International conference on machine learning*, pages 544–552. PMLR, 2019.

[3] C. Benjamins, T. Eimer, F. Schubert, A. Mohan, S. Döhler, A. Biedenkapp, B. Rosenhahn, F. Hutter, and M. Lindauer. Contextualize me - the case for context in reinforcement learning. 2023.

[4] C. Costen, M. Rigter, B. Lacerda, and N. Hawes. Planning with hidden parameter polynomial mdps. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pages 11963–11971, 2023.

[5] F. Doshi-Velez and G. Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, page 1432. NIH Public Access, 2016.

[6] G. Dulac-Arnold, N. Levine, D. J. Mankowitz, J. Li, C. Paduraru, S. Gowal, and T. Hester. An empirical investigation of the challenges of real-world reinforcement learning. 2020.

[7] T. Gupta, W. Gong, C. Ma, N. Pawlowski, A. Hilmkil, M. Scetbon, A. Famoti, A. J. Llorens, J. Gao, S. Bauer, et al. The essential role of causality in foundation world models for embodied ai. *arXiv preprint arXiv:2402.06665*, 2024.

[8] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. *arXiv preprint arXiv:1912.01603*, 2019.

[9] D. Hafner, T. Lillicrap, I. Fischer, R. Villegas, D. Ha, H. Lee, and J. Davidson. Learning latent dynamics for planning from pixels. In *International conference on machine learning*, pages 2555–2565. PMLR, 2019.

[10] D. Hafner, T. Lillicrap, J. Ba, and M. Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020. URL `https://openreview.net/forum?id=S1lOTC4tDS`.

[11] D. Hafner, T. P. Lillicrap, M. Norouzi, and J. Ba. Mastering atari with discrete world models. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=0oabwyZbOu`.

[12] D. Hafner, J. Pasukonis, J. Ba, and T. Lillicrap. Mastering diverse domains through world models, 2024. URL `https://arxiv.org/abs/2301.04104`.

[13] N. Hansen, X. Wang, and H. Su. Temporal difference learning for model predictive control. *arXiv preprint arXiv:2203.04955*, 2022.

[14] N. Hansen, H. Su, and X. Wang. Td-mpc2: Scalable, robust world models for continuous control. *arXiv preprint arXiv:2310.16828*, 2023.

[15] N. Hansen, H. Su, and X. Wang. TD-MPC2: Scalable, robust world models for continuous control. In *The Twelfth International Conference on Learning Representations*, 2024. URL `https://openreview.net/forum?id=Oxh5CstDJU`.

[16] K. Khetarpal, M. Riemer, I. Rish, and D. Precup. Towards continual reinforcement learning: A review and perspectives. *Journal of Artificial Intelligence Research*, 75: 1401–1476, 2022.

[17] R. G. Krishnan, U. Shalit, and D. Sontag. Deep kalman filters. *arXiv preprint arXiv:1511.05121*, 2015.

[18] OpenAI. Gpt-4 technical report, 2024. URL `https://arxiv.org/abs/2303.08774`.

[19] R. Rombach, A. Blattmann, D. Lorenz, P. Esser, and B. Ommer. High-resolution image synthesis with latent diffusion models, 2022. URL `https://arxiv.org/abs/2112.10752`.

[20] M. R. Samsami, A. Zholus, J. Rajendran, and S. Chandar. Mastering memory tasks with world models. *arXiv preprint arXiv:2403.04253*, 2024.

[21] V. Shaj. Learning world models with hierarchical temporal abstractions: A probabilistic perspective. *arXiv preprint arXiv:2404.16078*, 2024.

[22] V. Shaj, P. Becker, D. Büchler, H. Pandya, N. van Duijkeren, C. J. Taylor, M. Hanheide, and G. Neumann. Action-conditional recurrent kalman networks for forward and inverse dynamics learning. In *Conference on Robot Learning*, pages 765–781. PMLR, 2021.

[23] V. Shaj, D. Buchler, R. Sonker, P. Becker, and G. Neumann. Hidden parameter recurrent state space models for changing dynamics scenarios. *International Conference On Learning Representations*, 2022.

[24] V. Shaj Kumar, S. Gholam Zadeh, O. Demir, L. Douat, and G. Neumann. Multi time scale world models. *Advances in Neural Information Processing Systems*, 36: 26764–26775, 2023.

[25] R. S. Sutton. Reinforcement learning: An introduction. *A Bradford Book*, 2018.

[26] M. Towers, A. Kwiatkowski, J. Terry, J. U. Balis, G. De Cola, T. Deleu, M. Goulão, A. Kallinteris, M. Krimmel, A. KG, et al. Gymnasium: A standard interface for reinforcement learning environments. *arXiv preprint arXiv:2407.17032*, 2024.

[27] S. Tunyasuvunakool, A. Muldal, Y. Doron, S. Liu, S. Bohez, J. Merel, T. Erez, N. H. Timothy Lillicrap, and Y. Tassa. dm_control: Software and tasks for continuous control. *Software Impacts*, 6:100022, 2020. doi: https://doi.org/10.1016/j.simpa.2020.100022. URL `https://www.sciencedirect.com/science/article/pii/S2665963820300099`.

[28] L. van der Maaten and G. Hinton. Visualizing data using t-sne. *Journal of Machine Learning Research*, 9(86):2579–2605, 2008. URL `http://jmlr.org/papers/v9/vandermaaten08a.html`.

[29] A. Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[30] M. Volpp, F. Flürenbrock, L. Grossberger, C. Daniel, and G. Neumann. Bayesian context aggregation for neural processes. In *International Conference on Learning Representations*, 2021. URL `https://openreview.net/forum?id=ufZN2-aehFa`.

[31] A. Xie, J. Harrison, and C. Finn. Deep reinforcement learning amidst lifelong non-stationarity. *arXiv preprint arXiv:2006.10701*, 2020.

[32] A. Zhang, S. Sodhani, K. Khetarpal, and J. Pineau. Learning robust state abstractions for hidden-parameter block mdps. *arXiv preprint arXiv:2007.07206*, 2020.

A Objective Derivation

$$
\begin{aligned}
&\ln p\left(\boldsymbol{o}_{1:T}, r_{1:T} \mid \boldsymbol{a}_{1:T}, \boldsymbol{C_l}\right) \\
&= \ln \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[p\left(\boldsymbol{o}_{1:T}, r_{1:T} \mid \boldsymbol{a}_{1:T}, \boldsymbol{l}\right)\right] \quad \text{(Data aggregation for latent task inference given the context } \boldsymbol{C_l}\text{)} \\
&= \ln \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{p(\boldsymbol{s}_{1:T}|\boldsymbol{a}_{1:T},\boldsymbol{l})}{\mathbb{E}}\left[p\left(\boldsymbol{o}_{1:T}, r_{1:T} \mid \boldsymbol{s}_{1:T}, \boldsymbol{l}\right)\right]\right] \quad \text{(Posterior predictive log-likelihood using Latent Variable Model)} \\
&= \ln \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\int p\left(\boldsymbol{s}_{1:T} \mid \boldsymbol{a}_{1:T}, \boldsymbol{l}\right) p\left(\boldsymbol{o}_{1:T}, r_{1:T} \mid \boldsymbol{s}_{1:T}, \boldsymbol{l}\right)\, d\boldsymbol{s}_{1:T}\right] \quad \text{(Expectation definition)} \\
&= \ln \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\int \prod_{t=1}^{T} p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right)\, d\boldsymbol{s}_{1:T}\right] \quad \text{(Factorization due to Markov property)} \\
&= \ln \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\int \prod_{t=1}^{T} \frac{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)}{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)} p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right)\, d\boldsymbol{s}_{1:T}\right] \quad \text{(Variational approximate posterior)} \\
&= \ln \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q(\boldsymbol{s}_{1:T}|\boldsymbol{o}_{1:T},\boldsymbol{a}_{1:T},\boldsymbol{l})}{\mathbb{E}}\left[\prod_{t=1}^{T} \frac{p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right)}{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)}\right]\right] \quad \text{(Expectation w.r.t approximate posterior + exchange order of nominator factors)} \\
&\geq \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\ln \underset{q(\boldsymbol{s}_{1:T}|\boldsymbol{o}_{1:T},\boldsymbol{a}_{1:T}.\boldsymbol{l})}{\mathbb{E}}\left[\prod_{t=1}^{T} \frac{p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right)}{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)}\right]\right] \quad \text{(Jensen's inequality w.r.t. outer expectation)} \\
&\geq \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q(\boldsymbol{s}_{1:T}|\boldsymbol{o}_{1:T},\boldsymbol{a}_{1:T},\boldsymbol{l})}{\mathbb{E}}\left[\ln \prod_{t=1}^{T} \frac{p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right)}{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)}\right]\right] \quad \text{(Jensen's inequality w.r.t. inner expectation)} \\
&= \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q(\boldsymbol{s}_{1:T}|\boldsymbol{o}_{1:T},\boldsymbol{a}_{1:T},\boldsymbol{l})}{\mathbb{E}}\left[\sum_{t=1}^{T} \ln \frac{p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right)}{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)}\right]\right] \quad \text{(Product rule of logarithms)} \\
&= \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q(\boldsymbol{s}_{1:T}|\boldsymbol{o}_{1:T},\boldsymbol{a}_{1:T},\boldsymbol{l})}{\mathbb{E}}\left[\sum_{t=1}^{T} \ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) - \ln q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)\right]\right] \quad \text{(Quotient logarithm rule)} \\
&= \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q(\boldsymbol{s}_{1:T}|\boldsymbol{o}_{1:T},\boldsymbol{a}_{1:T},\boldsymbol{l})}{\mathbb{E}}\left[\sum_{t=1}^{T} \ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) + \ln p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) - \ln q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)\right]\right] \quad \text{(Product logarithm rule)} \\
&= \sum_{t=1}^{T} \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right) + \ln p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) - \ln q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)\right]\right] \quad \text{(Linearity of expectation)} \\
&= \sum_{t=1}^{T} \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right)\right]\right] + \\
&\qquad \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) - \ln q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)\right]\right] \quad \text{(Linearity of expectation)} \\
&= \sum_{t=1}^{T} \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right)\right]\right] + \\
&\qquad \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{\substack{q\left(\boldsymbol{s}_{t-1}|\boldsymbol{o}_{\leq t-1},\boldsymbol{a}_{<t-1},\boldsymbol{l}\right)\\ q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}}{\mathbb{E}}\left[\ln p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l}\right) - \ln q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t}, \boldsymbol{a}_{<t}, \boldsymbol{l}\right)\right]\right] \quad \text{(Expectation is constant for } t \notin \{t, t-1\}\text{)} \\
&= \sum_{t=1}^{T} \underset{p(\boldsymbol{l}|\boldsymbol{C_l})}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t, \boldsymbol{l}\right)\right]\right] +
\end{aligned}
$$

$$\underset{p(\boldsymbol{l}|\boldsymbol{C}_l)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_{t-1}|\boldsymbol{o}_{\leq t-1},\boldsymbol{a}_{<t-1},\boldsymbol{l}\right)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1},\boldsymbol{a}_{t-1},\boldsymbol{l}\right) - \ln q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)\right]\right]\right] \quad \text{(Expectation split)}$$

$$= \sum_{t=1}^{T} \underset{p(\boldsymbol{l}|\boldsymbol{C}_l)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t,\boldsymbol{l}\right)\right]\right] +$$

$$\underset{p(\boldsymbol{l}|\boldsymbol{C}_l)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_{t-1}|\boldsymbol{o}_{\leq t-1},\boldsymbol{a}_{<t-1},\boldsymbol{l}\right)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln \frac{p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1},\boldsymbol{a}_{t-1},\boldsymbol{l}\right)}{q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}\right]\right]\right] \quad \text{(Quotient rule for logarithms)}$$

$$= \sum_{t=1}^{T} \underset{p(\boldsymbol{l}|\boldsymbol{C}_l)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t,\boldsymbol{l}\right)\right]\right] +$$

$$\underset{p(\boldsymbol{l}|\boldsymbol{C}_l)}{\mathbb{E}}\left[\underset{q\left(\boldsymbol{s}_{t-1}|\boldsymbol{o}_{\leq t-1},\boldsymbol{a}_{<t-1},\boldsymbol{l}\right)}{\mathbb{E}}\left[D_{\mathrm{KL}}\left(q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right) \parallel p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1},\boldsymbol{a}_{t-1},\boldsymbol{l}\right)\right)\right]\right] \quad \text{(KL-Divergence definition)}$$

$$= \sum_{t=1}^{T} \underbrace{\underbrace{\underset{\substack{p(\boldsymbol{l}|\boldsymbol{C}_l),\\ q\left(\boldsymbol{s}_t|\boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right)}}{\mathbb{E}}\left[\ln p\left(\boldsymbol{o}_t, r_t \mid \boldsymbol{s}_t,\boldsymbol{l}\right)\right]}_{\textbf{Reconstruction}} + \underbrace{\underset{\substack{p(\boldsymbol{l}|\boldsymbol{C}_l),\\ q\left(\boldsymbol{s}_{t-1}|\boldsymbol{o}_{\leq t-1},\boldsymbol{a}_{<t-1},\boldsymbol{l}\right)}}{\mathbb{E}}\left[D_{\mathrm{KL}}\left(q\left(\boldsymbol{s}_t \mid \boldsymbol{o}_{\leq t},\boldsymbol{a}_{<t},\boldsymbol{l}\right) \parallel p\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1},\boldsymbol{a}_{t-1},\boldsymbol{l}\right)\right)\right]}_{\textbf{Regularization}}}_{\textbf{Context-dependent Evidence Lower Bound}}$$

B Algorithm and Implementation Details

B.1 Task Abstractions Via Bayesian Aggregation

Given a set of encoded interaction tuples and their corresponding variances $\{\boldsymbol{x}_n^l, \boldsymbol{\sigma}_n^l\}_{n=1}^N$, using the prior and observation model assumptions in Section 3.1 of the main paper, we infer the latent task abstraction $p\left(\boldsymbol{l} \mid \boldsymbol{C}_l\right) = \mathcal{N}(\boldsymbol{\mu}_l, \boldsymbol{\Sigma}_l) = \mathcal{N}(\boldsymbol{\mu}_l, \mathrm{diag}(\boldsymbol{\sigma}_l))$ as a Bayesian aggregation [30] of these using the following closed-form equations:

$$\boldsymbol{\sigma}_{\boldsymbol{l}}^2 = \left(\left(\boldsymbol{\sigma}_0^2\right)^{\ominus} + \sum_{n=1}^{N}\left(\left(\boldsymbol{\sigma}_n^{\boldsymbol{l}}\right)^2\right)^{\ominus}\right)^{\ominus},$$

$$\boldsymbol{\mu}_{\boldsymbol{l}} = \boldsymbol{\mu}_0 + \boldsymbol{\sigma}_{\boldsymbol{l}}^2 \odot \sum_{n=1}^{N}\left(\boldsymbol{x}_n^{\boldsymbol{l}} - \boldsymbol{\mu}_0\right) \oslash \left(\boldsymbol{\sigma}_n^{\boldsymbol{l}}\right)^2$$

where $\ominus$, $\odot$, and $\oslash$ denote element-wise inversion, product, and division, respectively.

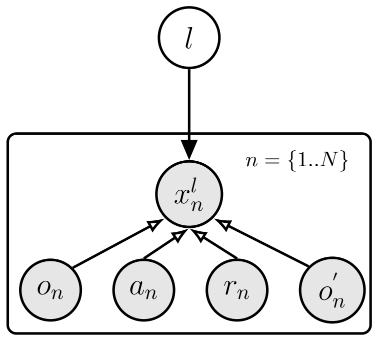


Figure 6: Generative model for the abstract latent task $\boldsymbol{l}$. The hollow arrows are deterministic transformations leading to implicit distribution $\boldsymbol{x}_n^l$ using a set encoder.

Discussion: This closed-form solution represents Bayesian aggregation, which can be interpreted as a form of **probabilistic attention**. The uncertainty $\boldsymbol{\sigma}_n^{\boldsymbol{l}}$ about each transition in the set encoder functions as attention weights, assigning higher weights to the most informative transitions. The derived update equations have only a linear computational complexity of $O(N)$, while similar deep set operations (self-attention) in transformers [29] have a complexity of $O(N^2)$.

B.2 HiP-Dreamer

Algorithm 1 Hidden Parameter Dreamer (HiP-Dreamer)

```
Require:
    Hyperparameters                         Model Parameters
    Seed episodes: $S$                      Context set encoder: $p_{\boldsymbol{\theta}}(\boldsymbol{l} \mid \boldsymbol{C_l})$ with
    Collect interval: $C$                   $\boldsymbol{C_l} = \{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o\prime})_t\}_{t=1}^{N}$
    Batch size: $B$                         Representation: $p_{\boldsymbol{\theta}}(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_t, \boldsymbol{l})$
    Context size: $N$                       Observation: $q_{\boldsymbol{\theta}}(\boldsymbol{o}_t \mid \boldsymbol{s}_t, \boldsymbol{l})$
    Sequence length: $L$                    Transition: $q_{\boldsymbol{\theta}}(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{l})$
    Imagination horizon: $H$                Reward: $q_{\boldsymbol{\theta}}(r_t \mid \boldsymbol{s}_t, \boldsymbol{l})$
    Learning rate: $\alpha$                 Actor: $\pi_{\boldsymbol{\phi}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{l})$
    Trajectory length: $T$                  Critic: $v_{\boldsymbol{\psi}}(\boldsymbol{s}_t, \boldsymbol{l})$
    Training epochs: $E$
    Action Repeat: $R$
1: Initialize dataset $\mathcal{D}$ with $S$ random seed episodes.
2: Initialize neural network parameters $\boldsymbol{\theta}$, $\boldsymbol{\phi}$, $\boldsymbol{\psi}$ randomly.
3: while not converged do
4:     for update step $c = 1$ to $C$ do
5:         // Dynamics learning
6:         Draw $\mathcal{B}$ data sequences $\{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o\prime})_t\}_{t=k}^{k+L} \sim \mathcal{D}$ uniformly at random from
7:         the dataset $\mathcal{D}$, with random start index k within an episode.
8:         Draw $\mathcal{B}$ consecutive context chunks $\boldsymbol{C_l} = \{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o\prime})_t\}_{t=k-N}^{k-1} \sim \mathcal{D}$
9:         of previous N interaction transitions.
10:        Infer latent task posterior $p_{\boldsymbol{\theta}}(\boldsymbol{l} \mid \boldsymbol{C_l})$ and sample $\boldsymbol{l}$ using reparameterization.
11:        Use the same latent task sample $\boldsymbol{l}$ for the entire sequence length L.
12:        Compute model states $\boldsymbol{s}_t \sim p_{\boldsymbol{\theta}}(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_t, \boldsymbol{l})$.
13:        Update model parameters using derived objective in Section 3.2.
14:        // Behavior learning
15:        Imagine trajectories $\{(\boldsymbol{s}_\tau, \boldsymbol{a}_\tau)\}_{\tau=t}^{t+H}$ from each $\boldsymbol{s}_t$ conditioned on derived task $\boldsymbol{l}$.
16:        Use the same latent task samples $\boldsymbol{l}$ for the entire imagine horizon H.
17:        Predict rewards $\mathbb{E}[q_{\boldsymbol{\theta}}(r_\tau \mid \boldsymbol{s}_\tau, \boldsymbol{l})]$ and values $v_{\boldsymbol{\psi}}(\boldsymbol{s}_\tau, \boldsymbol{l})$.
18:        Compute value estimates $V_\lambda(\boldsymbol{s}_\tau, \boldsymbol{l})$ via Equation 6 from [10].
19:        Update actor parameters $\boldsymbol{\phi} \leftarrow \boldsymbol{\phi} + \alpha \nabla_{\boldsymbol{\phi}} \sum_{\tau=t}^{t+H} V_\lambda(\boldsymbol{s}_\tau, \boldsymbol{l})$.
20:        Update critic parameters $\boldsymbol{\psi} \leftarrow \boldsymbol{\psi} - \alpha \nabla_{\boldsymbol{\psi}} \sum_{\tau=t}^{t+H} \frac{1}{2} \| v_{\boldsymbol{\psi}}(\boldsymbol{s}_\tau, \boldsymbol{l}) - V_\lambda(\boldsymbol{s}_\tau, \boldsymbol{l}) \|^2$.
21:    end for
22:    // Environment interaction
23:    $o_1 \leftarrow$ env.reset().
24:    Collect initial context chunk $\boldsymbol{C}_{\boldsymbol{l}_1}$ using random actions.
25:    for time step $t = 1$ to $\frac{T}{R}$ do
26:        Infer latent task posterior $p_{\boldsymbol{\theta}}(\boldsymbol{l}_t \mid \boldsymbol{C}_{\boldsymbol{l}_t})$ and sample $\boldsymbol{l}_t \sim p_{\boldsymbol{\theta}}(\boldsymbol{l}_t \mid \boldsymbol{C}_{\boldsymbol{l}_t})$.
27:        Compute $\boldsymbol{s}_t \sim p_{\boldsymbol{\theta}}(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_t, \boldsymbol{l}_t)$ from history.
28:        Compute $\boldsymbol{a}_t \sim \pi_{\boldsymbol{\phi}}(\boldsymbol{a}_t \mid \boldsymbol{s}_t, \boldsymbol{l}_t)$ with the action model.
29:        Add exploration noise to action $\boldsymbol{a}_t$.
30:        for action repeat $k = 1..R$ do
31:            $r_{t,k}, \boldsymbol{o}_{t,k+1} \leftarrow$ env.step($\boldsymbol{a}_t$)
32:        end for
33:        $r_t, \boldsymbol{o}_{t+1} \leftarrow \sum_{k=1}^{R} \gamma^{k-1} r_{t,k}, \boldsymbol{o}_{t,R+1}$
34:        Add experience to dataset $\mathcal{D} \leftarrow \mathcal{D} \cup \{(\boldsymbol{o}_t, \boldsymbol{a}_t, r_t, \boldsymbol{o}_{t+1})\}$.
35:        Update context chunk $\boldsymbol{C}_{\boldsymbol{l}_t}$ in a sliding-window manner using $\{(\boldsymbol{o}_t, \boldsymbol{a}_t, r_t, \boldsymbol{o}_{t+1})\}$.
36:    end for
37:    // Agent evaluation
38:    if T mod E == 0 then:
39:        Evaluate agent by calculating the expected return from X episodes.
40:    end if
41: end while
```

The HiP-Dreamer algorithm operates in **four stages**:

1. Dynamics learning phase (Lines 6-14): Batch of $\mathcal{B}$ consecutive trajectory data chunks $\{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o\prime})_t\}_{t=k}^{k+L} \sim \mathcal{D}$ is sampled uniformly at random from the replay buffer and used to train the world model, where k indicates a random start index uniformly sampled at random within the episode and is clipped to not exceed the episode length minus the training sequence length. Concurrently, $\mathcal{B}$ consecutive context chunks of N previous interaction transitions are drawn $\boldsymbol{C_l} = \{(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o\prime})_t\}_{t=k-N}^{k-1} \sim \mathcal{D}$. If fewer than N previous transitions are available, the context is zero-padded at the beginning. The context chunk $\boldsymbol{C_l}$ is used to infer the latent task posterior distribution $p_{\boldsymbol{\theta}}\left(\boldsymbol{l} \mid \boldsymbol{C_l}\right)$ in closed form using Equation 8 from [30]. To approximate the outer context-dependent expectation of the objective from Section 3.2, a reparameterized sample from the inferred latent task posterior distribution $\boldsymbol{l} \sim p_{\boldsymbol{\theta}}\left(\boldsymbol{l} \mid \boldsymbol{C_l}\right)$ is used, allowing gradients to flow through the sampling process. The latent task posterior $p_{\boldsymbol{\theta}}\left(\boldsymbol{l} \mid \boldsymbol{C_l}\right)$ is inferred once at the start of each training iteration and used for the entire sequence length L. All model components (including the context set encoder) are trained end-to-end using Backpropagation-Through-Time (**BPTT**).

2. Behavior learning phase (Lines 16-21): The actor chooses actions to predict imagined sequences of compact model states, while the critic accumulates the future predicted rewards beyond the planning horizon. Both actor and critic use learned model states, benefiting from the world model's representations. Each posterior state inferred during model training serves as an initial state for the actor's latent trajectory imagination. The latent task posterior $p_{\boldsymbol{\theta}}\left(\boldsymbol{l} \mid \boldsymbol{C_l}\right)$ remains consistent across the entire horizon length H = 15, a reasonable assumption for such short horizons. The actor aims to output actions that maximize the prediction of long-term future rewards made by the critic. The critic model is trained to regress the $\lambda -$ **returns**, computed as in Equation 8 from [10]. The world model (including the set encoder) is fixed during behavior learning, so the actor and critic gradients do not affect its representations.

3. Environment interaction phase (Lines 24-37): Initial context sequence chunk $\boldsymbol{C_{l_1}}$ is collected using random actions and updated in a sliding window manner, incorporating every new collected transition $(\boldsymbol{o}, \boldsymbol{a}, r, \boldsymbol{o\prime})_t$ to adapt to environmental changes as fast as possible. At each point in time, the context chunk $\boldsymbol{C_{l_t}}$ is used to infer the latent task posterior $\boldsymbol{l}_t \sim p_{\boldsymbol{\theta}}\left(\boldsymbol{l}_t \mid \boldsymbol{C_{l_t}}\right)$ and sample $\boldsymbol{l}_t$, followed by inferring an approximate latent state posterior $\boldsymbol{s}_t \sim p_{\boldsymbol{\theta}}(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_t, \boldsymbol{l}_t)$. An action trajectory is generated by conditioning the actor on the inferred latent state $\boldsymbol{s}_t$ and latent task $\boldsymbol{l}_t$, repeating the chosen action R times.

4. Evaluation phase (Lines 39-43): Every E epoch the agent's performance is assessed over X episodes to estimate the mean return. Unlike data collection, the evaluation phase uses the mean $\boldsymbol{\mu_{l_t}}$ of the latent task distribution $p_{\boldsymbol{\theta}}\left(\boldsymbol{l}_t \mid \boldsymbol{C_{l_t}}\right)$ enabling deterministic behavior, followed by approximating the latent state posterior $p_{\boldsymbol{\theta}}\left(\boldsymbol{s}_t \mid \boldsymbol{s}_{t-1}, \boldsymbol{a}_{t-1}, \boldsymbol{o}_t, \boldsymbol{\mu_{l_t}}\right)$, repeating the chosen action R times.

B.3 Hyperparameters.

This section details only those hyperparameter values that differ from the original architectures or correspond to new architectural components, such as the set encoder. For additional hyperparameters related to all algorithm stages not explicitly mentioned here, we refer to [10].

Set encoder. The set encoder includes a shared fully connected layer with 240 units, followed by two separate fully connected layers of 240 units each. One layer computes the latent task observation $\boldsymbol{x}_n^{\boldsymbol{l}}$, while the other computes the latent task variance $\boldsymbol{\sigma}_n^{\boldsymbol{l}}$. All set encoder layers use the ELU activation function. The latent task posterior $p_{\boldsymbol{\theta}}\left(\boldsymbol{l} \mid \boldsymbol{C}_{\boldsymbol{l}}\right)$ is modelled as a multivariate Gaussian with 20 dimensions. Due to computational and time constraints, extensive hyperparameter optimization was not conducted.

Learning updates. To train the world model, we use batches of 50 sequences of length 50, as in [8]. For the context data, we sample batches of 50 sequences from the replay buffer, each consisting of 20 prior interaction transitions. If fewer than 20 prior transitions are available, we use zero-padding to fill the context data before concatenating it with any available transition data. The set encoder updates use the same learning rate as the world model.

Objective and Critic. In our objective we use the KL-Balancing technique introduced in [11], which we found essential for stabilizing learning when using proprioceptive inputs. Additionally, we experienced instability problems w.r.t. critic network in many environments. To stabilize the critic training further, we use target network to calculate the critic targets, using a soft-update, realized as $\boldsymbol{\theta}_{t+1} = \tau\boldsymbol{\theta}_t + (1-\tau)\boldsymbol{\theta}_{t-1}$ with $\tau = 0.05$ for Gymnasium-based environments. For DMC multi-task benchmarks, we use a hard update, copying the critic's weights every 100 gradient steps.

C Evaluation Details

Despite the prevalence of non-stationarity in real-world environments and the existence of some non-stationary benchmarks such as [3] and [6], a detailed categorization of environmental changes remains absent, along with analyses of model-based reinforcement learning agent's adaptability across various non-stationary scenarios.

We categorize environmental changes into three primary types based on the temporal and structural aspects of the environment's evolution:

- **Dynamical Changes:** Alterations affecting the robot's system or the physical properties of the environment.
- **Objective Changes:** Modifications of objectives, such as changes in target velocity or task requirements.
- **Combined Changes:** Concurrent occurrences of multiple dynamical changes or a combination of dynamical and objective changes.

Each category of change can occur either inter-episodically (between episodes) or intra-episodically (within episodes). The temporal dimension specifies when changes occur, while the structural dimension identifies which environmental aspects are affected. Addressing these types of changes enables the design of robust model-based reinforcement learning agents capable of adapting to a broad range of environmental dynamics.

C.1 Dynamical Changes

We implemented various dynamical changes by modifying Gymnasium [26] and DMC Control Suite [27] environments, particularly focusing on Half Cheetah, Walker, and Hopper robots. The dynamical change values were chosen to ensure that MuJoCo's models remained valid while still challenging pre-trained model-free and model-based agents.

In environments with dynamical changes, the Oracle agent is conditioned on a vector that fully describes the changes. For instance, wind friction on Half Cheetah is represented by a vector of Cartesian forces acting on each body part. Joint perturbations are similarly represented, while actuator masking is indicated by a one-hot vector. In Hopper and Walker, the Oracle receives vectors representing body masses, inertia, and contact friction coefficients, respectively. Details for each environment are provided below:

Half Cheetah

- **Wind Friction**: Random 3D Cartesian forces are applied at the center of mass of each body part, directed either with or against the agent's movement, sampled from the range $[-10, 10]N$.
- **Joint Perturbation**: Random torques are added to joint torques resulting from the agent's actions, sampled from the range $[-20, 20]Nm$.
- **Actuator Deactivation**: One of the six actuators is randomly deactivated, meaning the agent's actions do not affect that actuator.

Hopper

- **Body Mass Inertia**: A random number of body parts is chosen that will be affected by the change and a random scaling factor from $[0.1, 2.5]$ is applied to both mass and inertia, resulting in varying weights and inertia for each affected part.

Walker

- **Contact Friction**: Friction coefficients at ground contact points are modified by randomly selecting new values from $[0.1, 3.9]$. The default feet friction coefficient is 1.9.

The environmental changing values are sampled always randomly from the predefined value ranges. For these types of dynamic changes, a vanilla agent may still adapt because the optimal behavior's state region remains the same, what changes is how the agent reaches it.

C.2 Objective changes

Half Cheetah, Hopper, and Walker

- Target Velocity: A target velocity is randomly chosen within the range $[-6, 6]\frac{m}{s}$.

In these environments, the Oracle agent receives the target velocity as a scalar.

DMC Multi-Task Benchmarks

- Objective Change: At the beginning of each episode, a new objective is selected, requiring the agent to acquire multiple skills within the same domain.

We implement these multi-task benchmarks using pre-defined tasks from [15], creating a variety of benchmarks across different domains and categorizing them by task count, as shown in Table 1.

In all DMC multitask environments, the Oracle agent receives a unique one-hot vector corresponding to each environment.

Table 1: Multi-task benchmarks based on Cheetah, Walker, Ball-In-Cup, and Pendulum models.

Cheetah Tasks	Walker Tasks	Cup Tasks	Pendulum Tasks
Stand Front	Walk Backwards	Catch	Swingup
Stand Back	Run Backwards	Spin	Spin
Run Front	Walk		
Run Back	Run		
Run Backwards	Walk Backwards		
Stand Front	Arabesque		
Stand Back	Lie Down		
Jump	Legs Up		
Run Front	Head Stand		
Run Back	Walk		
Lie Down	Flip		
Legs Up	Backflip		
Flip			
Flip Backwards			

C.3 Further Evaluation Under Dynamical Changes

This section presents additional evaluations of environments with both inter-episodic and intra-episodic dynamical changes.

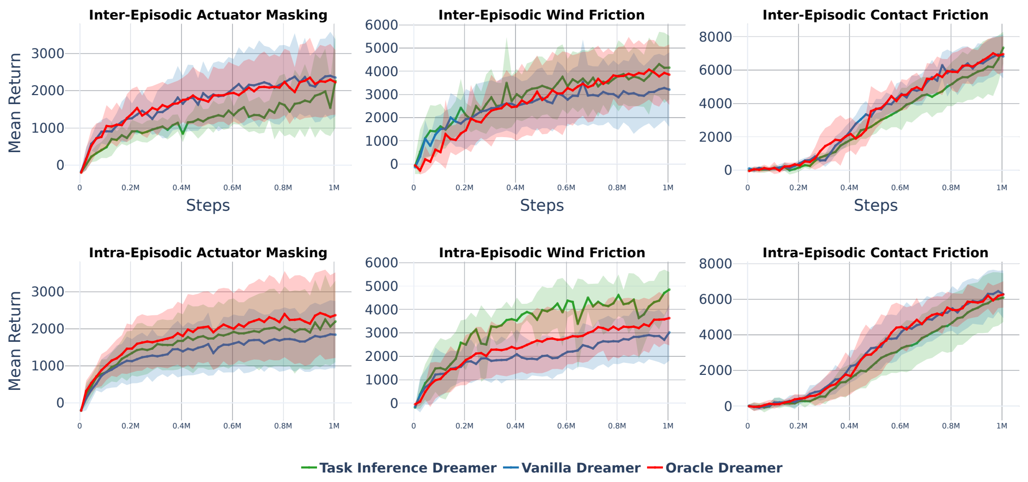


Figure 7: Dynamical changes on HalfCheetah and Walker. The first row shows inter-episodic dynamical changes, whereas the second row shows intra-episodic dynamical changes with a frequency of 200 environmental steps.

Figure 7 illustrates additional experiments involving dynamical changes in the environment. Across both inter- and intra-episodic changes, task-conditioned agents exhibit similar performance to the vanilla agent, with only minor differences observed.

On one hand, this result suggests that the vanilla agent is capable of adapting to various dynamic changes, indicating that approaches based on the POMDP formalism can indeed learn representations that support adaptive behavior. On the other hand, the HiP-POMDP formalism generally performs comparably or slightly better, highlighting the benefits and expressiveness of a learned latent task representation.

Interestingly, in some cases, the task-inference agent even outperforms the Oracle agent, despite the former having to learn and utilize a task representation for each task, while the Oracle agent is directly provided with task change information. This advantage may be due to the learned task representation capturing additional task-relevant information beyond the task changes alone, or it could suggest that the ground-truth task representation provided to the Oracle agent is not optimally structured for adaptation. This observation warrants further investigation in future work.

C.4 Further Evaluation Under Objective Changes

In this section, we conduct additional experiments under various reward/objective changes and identify the points at which different algorithms begin to fail.

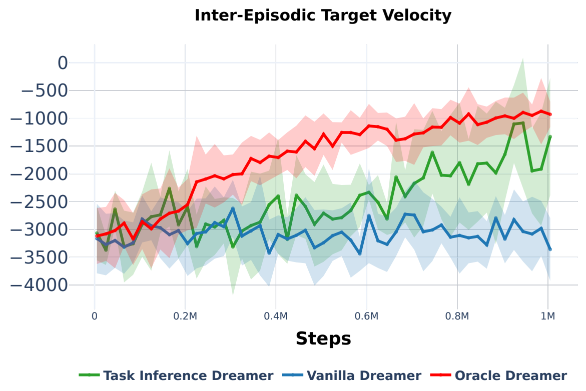


Figure 8: Inter-Episodic target velocity change on Hopper.

Figure 8 provides further empirical evidence that the vanilla agent struggles to handle objective changes, while task-conditioned agents demonstrate adaptive behavior, consistent with the results shown in Figure 4.

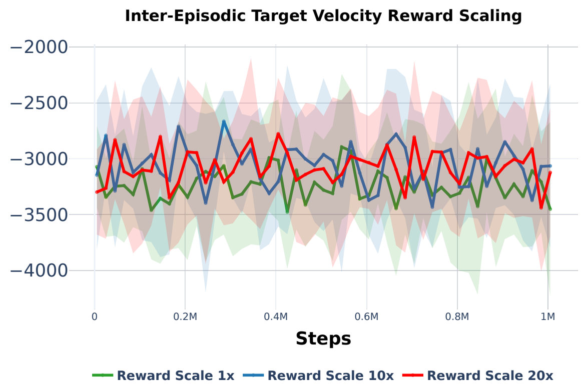


Figure 9: Vanilla Dreamer agent under inter-episodic reward change on Half Cheetah with different reward loss scaling factors.

Breaking point of Vanilla Dreamer under objective changes. To investigate the vanilla agent's failures under target reward-changing experiments, we adjust the reconstruction loss by giving more weight to reward reconstruction to offset potential higher loss weights from the multi-dimensional observations. Figure 9 illustrates the effect of scaling the reward reconstruction loss differently.

Our observations show that increasing the reward reconstruction weight factor also raises the KL divergence between the prior and posterior distributions over the latent state, along with an increased observation reconstruction loss, ultimately degrading performance. These findings suggest the optimization process struggles to balance reconstruction and regularization loss terms, possibly leading to overfitting to specific rewards within each mini-batch.

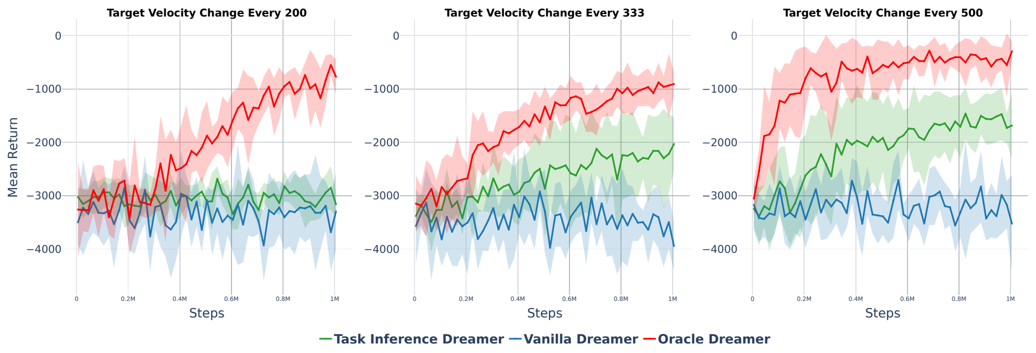


Figure 10: Intra-episodic target velocity change on Half Cheetah. The change frequency is reduced from left to right.

Breaking point of Task-inference Dreamer under objective changes. Figure 10 highlights a breaking point in the latent task inference mechanism when the target velocity changes every 200 environmental steps. We hypothesize that the standard latent task aggregation requires more time to infer a new task belief accurately, a necessity for effective adaptation, especially when target velocity undergoes drastic changes near boundary values. To address this, we are currently experimenting with a modified latent task update mechanism that reduces the influence of older transitions, introducing a "forgetting" effect that could help the agent adapt more rapidly to abrupt task shifts.

DMC-Multitask benchmarks We also assess all agents under more complex objective change settings, where each agent must learn multiple skills simultaneously. Figure 11 shows results on a variety of different custom-designed multi-task benchmarks.

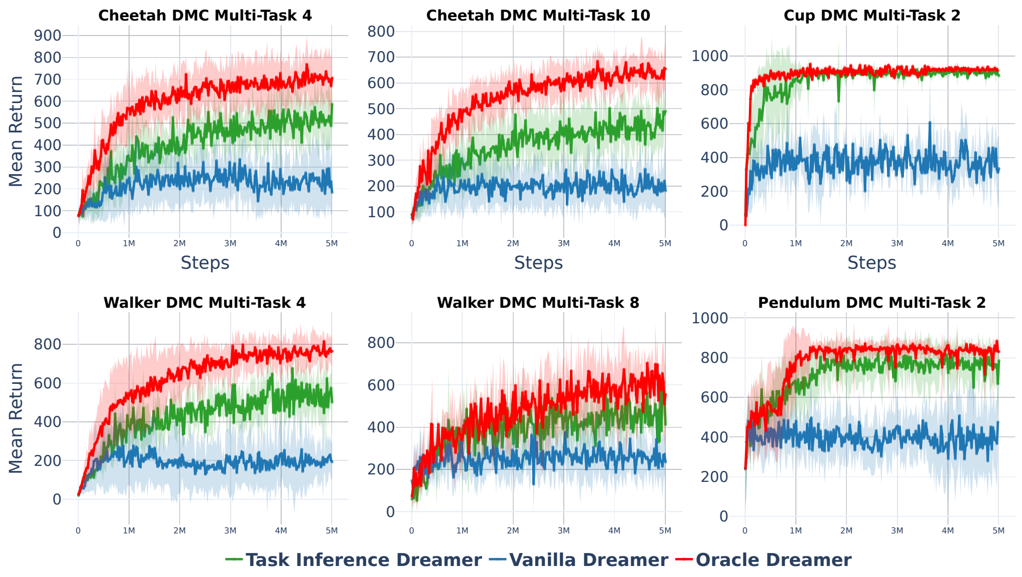


Figure 11: Inter-episodic objective changes on DMC Multi-task benchmarks on Half Cheetah, Walker, Cup, and Pendulum domains. Each number indicates the number of tasks in each experiment.

The vanilla agent struggles to learn multiple skills simultaneously. We hypothesize this limitation arises from multiple interferences in the learned latent state, resulting in a task-independent latent space structure. In such cases, the reward predictor cannot estimate rewards accurately, leading to suboptimal performance. However, providing task information enables better multi-skill learning across all benchmarks, potentially creating a more structured latent space. Additional evidence for these hypotheses is presented in Appendix C.6.

C.5 Evaluation under Combined Environmental Changes

In this section, we evaluate all agents in the most challenging setting, involving both multiple dynamical changes and combined dynamical and objective changes. Figure 12 illustrates the evaluation results for all agents on the Half Cheetah task, where changes evolve in an inter-episodic manner.

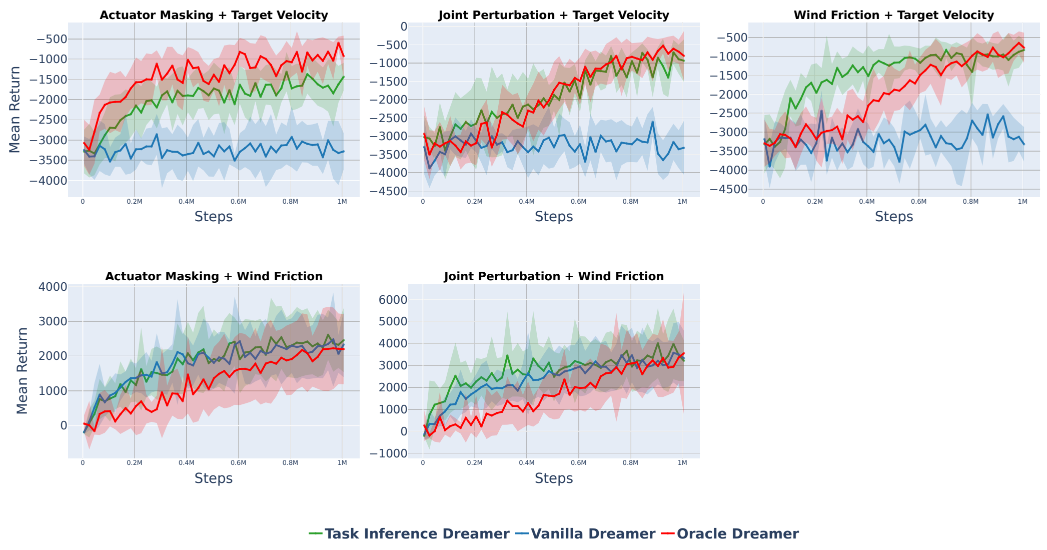


Figure 12: Multiple changes on HalfCheetah. The first row shows combined dynamical and objective changes, whereas the second row shows combined dynamical changes.

As seen in Figure 12, the vanilla agent demonstrates adaptability under multi-modal dynamical changes, however, it struggles with combined dynamical and objective changes, largely due to its limited ability to handle shifts in objectives.

Introducing a task representation notably enhances performance, as both the task-inference and oracle agents successfully adapt across all combined change scenarios. However, under scenarios involving only combined dynamical changes, the oracle agent exhibits slower learning compared to the vanilla agent. This slower adaptation raises questions about the potential influences of the task representation, warranting further investigation.

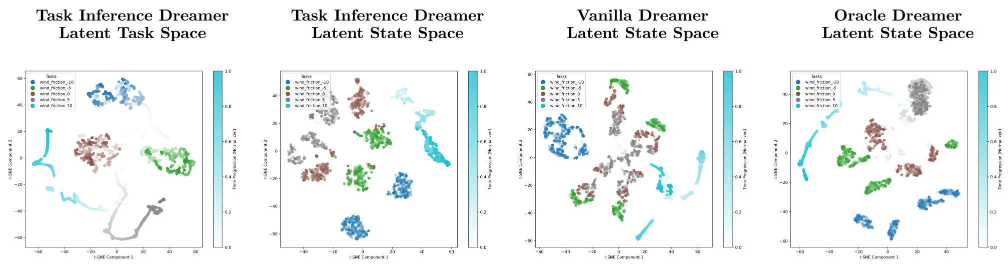
C.6 Latent Space Visualizations

This section presents 2D projections of both the world model's latent state and the latent task spaces. The visualizations are generated by recording trajectories of posterior latent states (incorporating both deterministic and stochastic components) and latent task representations during evaluations across various tasks, with a final 2D projection achieved using t-SNE [28].

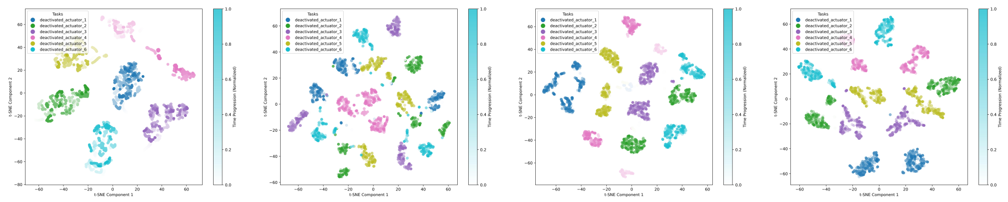
The primary objective of these visualizations is to explore two key questions: (1) Does conditioning the MBRL agent on either a ground truth or learned task representation introduce any structural changes in the latent state space? (2) How does the structure of the latent state space correlate with the agent's performance?

C.6.1 Dynamics Changes

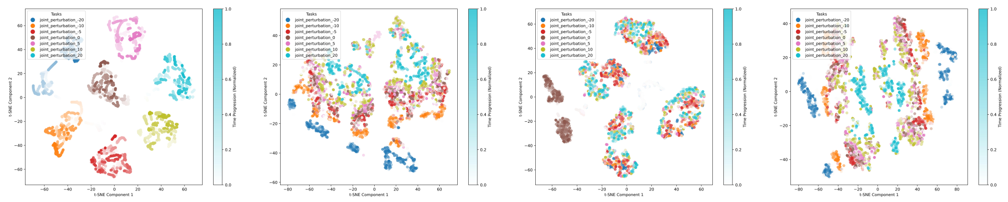
Figure 13 displays 2d projections of the latent state space under different dynamic environmental conditions.



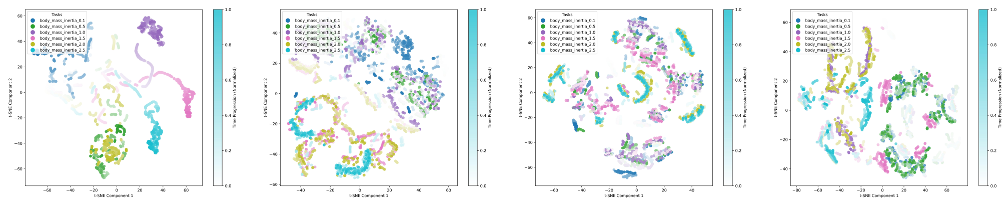
(a) Half Cheetah: wind friction.



(b) Half Cheetah: actuator masking.



(c) Half Cheetah: joint perturbation.



(d) Hopper: body mass and inertia.

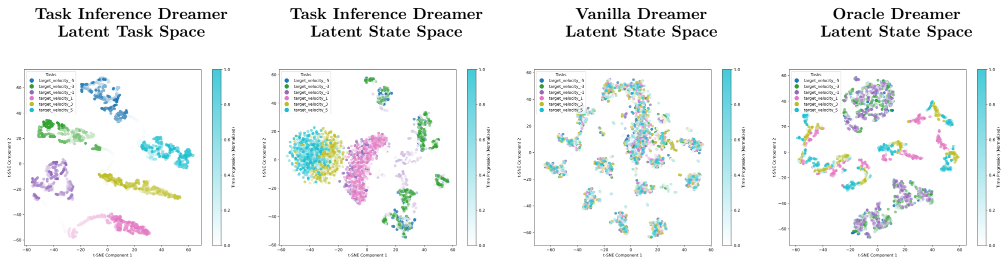
Figure 13: Comprehensive latent state and task space visualizations across various environments experiencing inter-episodic dynamical changes.

In the visualizations, the vanilla agent—grounded in the POMDP formalism—demonstrates task-dependent latent space structuring under different dynamic changes. Examining these projections along with empirical results from Figure 7 reveals a positive correlation between the structured latent space and the agent's performance. Notably, conditioning the agent on task representations produces an even more task-specific latent space structure.

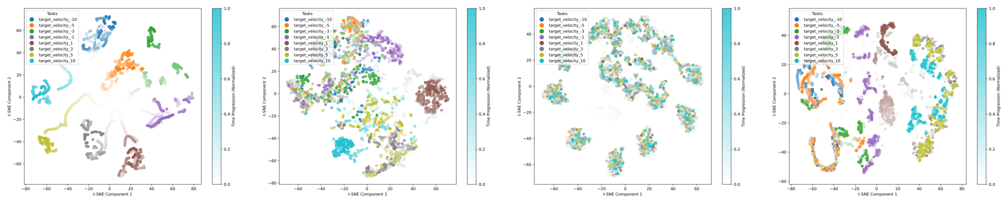
Task conditioning appears to provide two main advantages: (1) a more structured latent space enables enhanced data modeling, as task-specific representations minimize interferences, thus improving data reconstruction; and (2) task conditioning helps disambiguate overlapping latent states, especially when a state recurs across tasks. This disambiguation contributes to better data modeling and aids in achieving adaptive behaviors.

C.6.2 Objective Changes

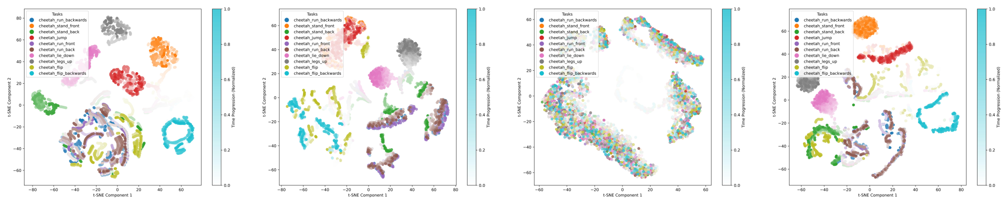
Figure 14 presents 2D projections of latent task and state spaces during evaluations in reward-altered environments, where the objective is to reach various target velocities or master multiple skills.



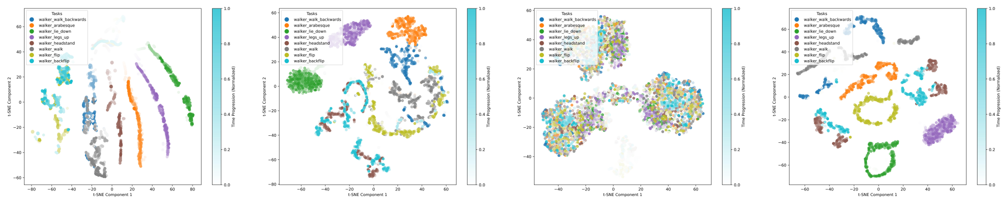
(a) Cheetah: target velocity.



(b) Walker: target velocity.



(c) Cheetah: 10 skills.



(d) Walker: 8 skills.

Figure 14: Comprehensive latent state and task space visualizations for agents learning to adapt to varying objectives, including different target velocities or skills.

Figure 14 reveals two critical observations. When the reward's structure changes, the vanilla agent's latent space does not organize itself by task, leading to substantial interference and making it challenging for all agent's components to infer task information, contributing to suboptimal performance as evidenced in Figure 11 for changing rewards.

In contrast, conditioning the agent on task representations produces a latent space that is not only more task-aware but also better suited for concurrent multi-task learning. Similarly to the findings in Section C.6.1 under dynamics changes, a positive correlation emerges between the structured latent space and the agent's final performance. However, for the most challenging skill-learning experiments, the task inference approach organizes latent space with some tasks represented jointly, which can hinder unique task identification and contribute to the observed performance gap in skill-learning tasks as shown in Figure 11.