Learning to Explore by Predicting What Matters: Entity-Centric Exploration for Visual RL

Anonymous authors
Paper under double-blind review

Abstract

Exploration is central to reinforcement learning, but it remains challenging in long-horizon tasks with high-dimensional visual inputs. In many such tasks, the most important structure is not captured at the level of raw pixels or full states, but at the level of entities and their interactions. In this work, we explore how entity structure can be leveraged to make exploration more efficient under visual observations. By establishing a formal connection between dynamics-prediction error and count-based exploration through the lens of information gain, we identify a core reduction in the sample efficiency for environments with certain entity-reward structures. Inspired by this, we introduce ECLAIR, an entity-centric exploration method that learns unsupervised, entity-centric representations online and uses a latent dynamics model to estimate entity-level novelty. By training online, ECLAIR allows the entity abstraction to improve as the agent encounters new states. We evaluate ECLAIR on challenging sparse-reward visual RL tasks and demonstrate that its exploration efficiency outperforms prior exploration baselines across skill-based navigation, dynamic control, and robotic manipulation. Additional visualizations and video rollouts are available on our anonymous website: https://sites.google.com/view/eclair-rl.

1 Introduction

Reinforcement learning has made impressive strides across a wide range of domains, from games and logistics optimization to robotics and large language models (Silver et al., 2017; Rolf et al., 2023; Guo et al., 2025; Li et al., 2026). Exploration plays an essential role in online reinforcement learning, as the key to policy improvement lies in collecting diverse experiences that reflect the environment dynamics or reward structure. However, not all experiences contribute equally to learning. Consider the task of a robot exploring a cluttered room: it must first explore how the objects around it relate to one another before it can make progress toward any goal and receive a positive reward. Simply moving randomly in free space fails to induce key states for learning, even though it might produce high novelty. In contrast, exploration strategies that reason about the entities in the scene can guide the agent toward meaningful transitions that lead to success. This raises a natural question:

How can we leverage the entity structure of the environment to achieve more efficient exploration?

A first challenge is to extract entities from high-dimensional observations. In many real-world tasks, entity-level abstractions can drastically reduce the effective state space. For instance, in the cluttered room example, representing the scene as $\{\texttt{robot}, \texttt{drawer}, \texttt{shirt}, \ldots\}$ is far more compact than reasoning directly over raw pixels. However, such structured abstractions are not always provided in practice. Moreover, the appropriate notion of an entity may evolve as the agent explores: objects that may appear as background early in training can be disentangled as the agent explores their manipulation. We therefore require an entity representation that can be learned from pixels and refined online. To this end, we build on Deep Latent Particles (DLP) (Daniel & Tamar, 2022), which discovers entities in an unsupervised manner.

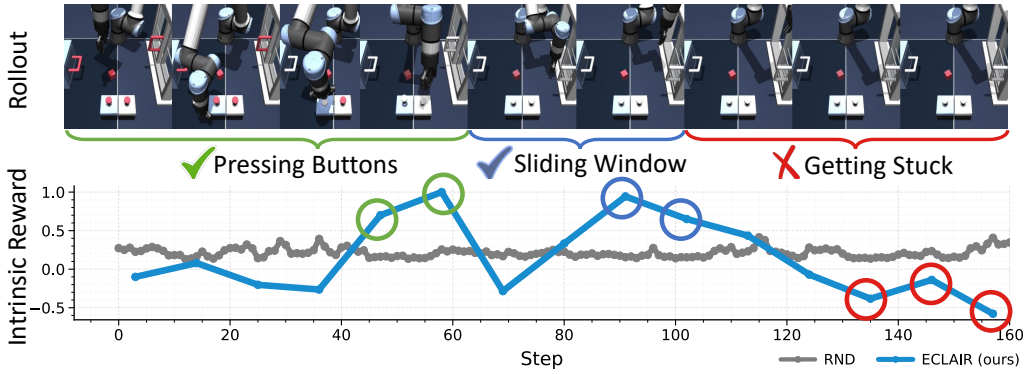


Figure 1: ECLAIR is an exploration method that assigns high intrinsic reward to meaningful object-state changes and low or negative reward to unproductive behavior. Compared to RND (Burda et al., 2018), ECLAIR leverages entities to direct exploration towards dynamically relevant states.

Prior exploration methods are grounded in the principle of optimism in the face of uncertainty: an agent should prefer states it has visited least. UCB (Auer et al., 2002) assigns bonuses inversely proportional to visit counts, which is theoretically optimal in the bandit setting. In high-dimensional settings where exact counts are intractable, Stadie et al. (2015) uses dynamics prediction error as a proxy for state novelty, as a model yields high error in rarely visited states. Bellemare et al. (2016) later formalized this intuition by showing that a learned density model can serve as a pseudo-count. However, the connection between the dynamics prediction error and pseudo-counts has not been formally established. To our knowledge, this work is the first to formalize this connection explicitly and further extend it to the factored setting by using per-entity dynamics prediction error as a structured estimator for pseudo-counts over the entity state space.

A separate class of exploration methods such as ELDEN (Hu et al., 2023) and Null (Chuck et al., 2025) targets entity-centric domains by leveraging interaction-encouraging principles. However, these methods lack the principled theoretical grounding of count-based approaches and have only been evaluated on low-dimensional state spaces. We extend the theory of count-based exploration to factorized state spaces. Rather than estimating pseudo-counts over the full joint state space, we decompose each state into K entity representations and use the *sum* of per-entity dynamics prediction errors as a structured estimator of novelty. This factored uncertainty estimate is more tractable than estimating uncertainty over the full joint space $|\mathcal{E}|^K$, and introduces an inductive bias that prioritizes changes in individual entities and their interactions. Together, this demonstrates that in entity-centric environments, exploration using dynamic prediction errors for factors can reduce the space from $\Theta(|\mathcal{E}|^K)$ to $\Theta(K|\mathcal{E}|)$. In practice, algorithms which leverage this structure have more efficient exploration and improves learning in long-horizon tasks.

We present our method ECLAIR, which stands for **E**ntity-**C**entric **L**atent dyn**A**mics **I**ntrinsic **R**eward. ECLAIR leverages a latent particle dynamics model built on DLP representations (Daniel & Tamar, 2022; Daniel et al., 2026), and uses the sum of per-entity KL divergences between the encoder posterior and the dynamics prior as a structured exploration bonus. Rather than keeping a fixed representation, we jointly train the entity encoder, the dynamics model, and the RL policy online, allowing the representations to improve as the agent collects more informative experiences. We evaluate ECLAIR on 11 challenging visual RL tasks, where it achieves an average score of 0.84 compared to 0.40 for the strongest baseline, substantially improving exploration across both navigation and robotic manipulation domains.

2 Background

Problem Setup. We consider a partially observed Markov Decision Process (POMDP) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{T}, \Omega, r, \gamma)$, where $s_t \in \mathcal{S}$ is the latent environment state, $a_t \in \mathcal{A}$ is the action, $o_t \in \mathcal{O}$ is

the observation, $\mathcal{T}$ is the transition dynamics, $\Omega(o_t \mid s_t)$ is the observation model, r is the extrinsic reward, and γ is the discount factor. The agent observes images o_t rather than state s_t. In sparse-reward settings, the extrinsic reward is zero for most transitions, making exploration challenging. A common approach is to augment the environment reward with an intrinsic bonus that encourages the agent to explore: $\tilde{r}_t = r_t^e + \beta r_t^i$, where $\beta > 0$ controls the strength of the intrinsic signal.

Count-Based Exploration and Prediction Error. Count-based exploration such as UCB (Auer et al., 2002) assigns larger bonuses to rarely visited states, commonly using bonuses of the form

$$r_t^i \propto \frac{1}{\sqrt{N_t(s_t)}},$$

where $N_t(s_t)$ denotes the number of visits to state s_t by time t. In high-dimensional or continuous spaces, exact state counts are unavailable, and pseudo-count methods Bellemare et al. (2016) replace $N_t(s_t)$ with a density-based estimate $\hat{N}_t(s_t)$. This perspective motivates using model uncertainty or prediction error as a proxy for novelty. In Section 4.1, we make this connection explicit by relating latent dynamics prediction error to pseudo-count-based intrinsic rewards. Additional background on count-based exploration is provided in Appendix B.

Factored Latent State Representations. Although the environment is partially observed from pixels, many domains admit a compact factored latent representation. We assume that the learned latent state can be represented as a set of K entity variables, each with latent dimension d:

$$z_t = \{e_t^1, e_t^2, \dots, e_t^K\}, \qquad e_t^k \in \mathbb{R}^d,$$

where e_t^k denotes the representation of the k-th entity at time t. Since this factored state is not directly observed, it can be inferred from the image observation using an object-centric encoder, $z_t = q_\phi(o_t) = \{e_t^1, \dots, e_t^K\}$.

Latent Particle World Model (LPWM). LPWM (Daniel et al., 2026) is a self-supervised variational autoencoder (VAE (Kingma & Welling, 2014)) for end-to-end object-centric world modeling from pixels. LPWM differs from monolithic latent world models such as Dreamer (Hafner et al., 2020a;b), which use a single global latent vector to represent the entire scene, whereas LPWM represents each image as a set of low-dimensional deep latent particles (DLP (Daniel & Tamar, 2022; 2024)). The model uses an autoregressive Transformer-based dynamics module to predict future latent states, optionally conditioned on actions, language, or goal images. Further details on the LPWM/DLP architecture and latent particle parameterization are provided in Appendix B.

3 Related Work

This work proposes a self-supervised, entity-centric exploration framework for visual RL. In the following paragraphs, we situate our approach within three key areas of prior research: count-based exploration, factored exploration, and entity-centric representation learning for decision making.

Count-based exploration. UCB (Auer et al., 2002; Jamieson et al., 2014) was originally introduced in the bandit setting, where it enjoys strong theoretical guarantees by implementing the principle of optimism in the face of uncertainty. Bellemare et al. (2016) formalizes this principle for deep RL by showing that pseudo-counts derived from a learned density model yield principled exploration bonuses, providing a tractable substitute for exact visitation counts in high-dimensional spaces. Subsequent works scale this idea further using random network distillation (Burda et al., 2018), coin-flipping counts (Lobel et al., 2023), and video prediction likelihoods (Escontrela et al., 2023) as uncertainty proxies. Our work instead uses dynamics prediction error (sometimes described as "curiosity") as the exploration signal, following Stadie et al. (2015) and Pathak et al. (2017). To our knowledge, we are the first to draw an *explicit* connection between dynamics prediction error and pseudo-count bonuses.

Exploration in factored MDPs. A complementary line of work studies exploration in factored MDPs, where the state is decomposed into interacting entities and the goal is to discover new

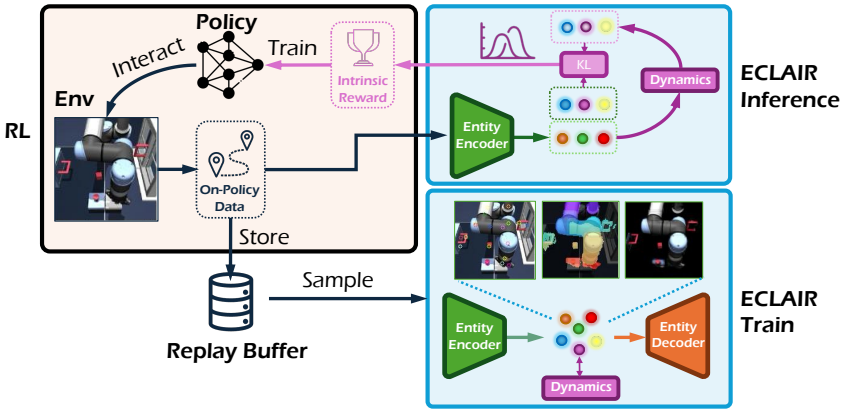


Figure 2: Overview of ECLAIR. ECLAIR learns an entity-centric representation and latent dynamics model online from the agent's experience. During inference, image observations are encoded into entity latents, and the KL divergence between the encoded posterior and the dynamics prediction is used as an intrinsic exploration reward. As the agent collects new experience, observations are stored in a replay buffer and sampled to continually update both the entity encoder and dynamics model.

interactions among them. Seitzer et al. (2021) measure local dependencies between action and environment entities via conditional mutual information. Sancaktar et al. (2022) measures novelty through ensemble disagreement. ELDEN (Hu et al., 2023) uses gradient-based partial derivatives of a learned dynamics model to identify uncertain entity dependencies, while Null (Chuck et al., 2025) employs counterfactual tests to detect meaningful entity interactions. However, all of these methods are evaluated exclusively in state-based environments with ground-truth factors, limiting their applicability to visual domains. FOCUS (Ferraro et al., 2025) learns and uses the entropy of latent object representations as an exploration signal but assumes ground-truth object segmentation. These works implicitly assume that their representations are sufficiently informative from offline training. In contrast, our work learns entity-centric representations in a fully self-supervised manner from pixel observations and updates the representation with newly collected experiences.

Entity-centric representations for RL. Prior works have demonstrated that entity-centric representations lead to improved performance and compositional generalization in visual RL. Zadaianchuk et al. (2021) pretrains a SCALOR encoder offline for goal-conditioned multi-object manipulation. ECRL (Haramati et al., 2024), EC-Diffuser (Qi et al., 2025), and HECRL (Haramati et al., 2026) leverage DLP representation (Daniel & Tamar, 2022) and demonstrate compositional generalization in robotic manipulation via Transformer-based architectures. SOLD (Mosbach et al., 2025) and OC-STORM (Zhang et al., 2025) learn object-centric dynamics from pixels for model-based RL, with the latter requiring annotated segmentations. All of these methods assume a sufficient representation can be learned from random interactions or offline data prior to RL training, and many use shaped rewards. In contrast, we tackle sparse reward problems by using an object-centric dynamics to guide exploration, while continually updating the representation with online samples collected by the policy.

4 Method

We now introduce our method, **ECLAIR** (**E**ntity-**C**entric **L**atent dyn**A**mics **I**ntrinsic **R**eward). We first introduce the theoretical motivation in Section 4.1. We then describe the practical instantiation of the model in Section 4.2 and the online updates with RL training in Section 4.3.

4.1 Theoretical Motivation

Dynamics prediction error and pseudo-counts. We first formalize the relationship between dynamics prediction error and pseudo-counts following the framework of Bellemare et al. (2016). Given (s, a), we identify the density model with the learned latent dynamics prior $\rho_n(z') := p_\xi(z' \mid z, a)$, and define the recoding probability as $\rho'_n(z') := p_{\xi'_n}(z' \mid z, a)$, where ξ'_n are the parameters

after an update toward the encoder posterior $q_\phi(z' \mid s')$. Rather than estimating pseudo-counts in the raw state space $\mathcal{S}$, we estimate them in the learned latent space $\mathcal{Z}$ induced by q_ϕ, which generalizes across states with similar dynamics and inherits the generalization property of pseudo-counts highlighted by Bellemare et al. (2016).

Assumption 1 (Gaussian Dynamics with Online Updates)**.** *Both the encoder posterior and dynamics prior are parameterized as isotropic Gaussians:*

$$q_\phi(z' \mid s') = \mathcal{N}(z';\, \mu_\phi(s'),\, \sigma_\phi^2(s')I), \quad p_\xi(z' \mid z, a) = \mathcal{N}(z';\, \mu_\xi(z,a),\, \sigma_\xi^2(z,a)I).$$

The dynamics prior is updated online with a small step size $\alpha \ll 1$ toward the encoder posterior.

Proposition 1 (Pseudocounts and Dynamics Prediction Error)**.** *Under Assumption 1, The pseudo-count and intrinsic reward are approximately:*

$$\hat{N}_n(s') \approx \frac{\sigma_\xi^2}{2\alpha \left\|\mu_\phi(s') - \mu_\xi(z,a)\right\|^2}, \tag{1}$$

$$r^i(s,a) = \frac{1}{\sqrt{\hat{N}_n(s')}} \propto \frac{\left\|\mu_\phi(s') - \mu_\xi(z,a)\right\|}{\sigma_\xi}. \tag{2}$$

In practice, we use $\mathrm{KL}(q_\phi(z' \mid s') \| p_\xi(z' \mid z, a))$ as the intrinsic bonus, as it is proportional to the squared prediction error under Assumption 1 and corresponds to the information gain $\mathrm{IG}_n(s')$. The full derivation is in Appendix A.1.

Entity-centric count reduces exploration complexity exponentially. After establishing a formal connection between dynamics prediction error and pseudo-counts, we further show that estimating entity-centric counts improves sample efficiency of UCB in the bandit setting, as a simplified abstraction of the exploration problem in MDP.

Let K denote the number of entities, $E = |\mathcal{E}_k|$ the number of states per entity, and $A = |\mathcal{A}|$ the number of actions. Following Jamieson et al. (2014), the UCB sample cost per arm is $\Theta(\Delta_i^{-2} \log\log \Delta_i^{-2})$, where Δ_i is the suboptimality gap. A monolithic contextual bandit has $\Theta(E^K \cdot A)$ such arms. We show that additive reward structure reduces this to $\Theta(KEA)$ arms.

Assumption 2 (Additive Reward Structure)**.** $\mathbb{E}[r(s,a)] = \frac{1}{K}\sum_{k=1}^{K} r_k(e^k, a)$, $r_k : \mathcal{E}_k \times \mathcal{A} \to \mathbb{R}$.

Proposition 2 (Factored UCB Complexity)**.** *Under Assumption 2, let $\Delta_k(e,a)$ denote the suboptimality gap of action a for entity k in state e, the UCB sample complexity is:*

$$N_{\text{fact}} = \sum_{k=1}^{K} \sum_{e \in \mathcal{E}_k} \sum_{a:\, \Delta_k(e,a) > 0} \Theta\big(\Delta_k(e,a)^{-2} \log\log \Delta_k(e,a)^{-2}\big)\,, \tag{3}$$

with a reduction to $\Theta(KE)$ arms versus $\Theta(E^K)$ in the monolithic case. The proof is in Appendix A.2.

Together, these results justify our using entity-centric prediction error as a practical proxy for pseudo-count exploration. While the assumptions are idealized, they capture the compositional structure present in many domains, where task progress depends on discovering which entities are relevant rather than exploring the full pixel space. We next describe our practical algorithm for visual RL.

4.2 Entity-Centric Exploration via Latent Dynamics Prediction

We now introduce the practical instantiation of ECLAIR, which uses the prediction error of a latent entity-centric dynamics model as an intrinsic exploration signal. Concretely, we build ECLAIR on LPWM (Daniel et al., 2026), which jointly learns a factored state representation $q_\phi(o_t) = \{e_t^1, \ldots, e_t^K\}$ from a pixel observation o_t together with an autoregressive dynamics prior over these entity latents. This end-to-end formulation is important: rather than pretraining a representation and

then fitting a separate dynamics model, LPWM trains both components jointly so that the latent entities are directly shaped to be predictable by the dynamics model.

The dynamics module defines an autoregressive prior $p_\xi(e_t \mid e_{t-1}, \ldots, e_0, a_{t-1})$, which predicts the distribution over next-step entity states conditioned on the entity history and the action history:

$$f_\xi\Big(\{\{e_t^k\}_{k=1}^K\}_{t=0}^{T-1},\ \{a_t\}_{t=0}^{T-1}\Big) = \{\{\hat{e}_t^k\}_{k=1}^K\}_{t=1}^{T}.$$

The dynamics module is implemented as a causal spatio-temporal Transformer, where a separate context module maps global action conditioning to per-entity conditioning signals applied in the dynamics module using Adaptive Layer Normalization (AdaLN (Peebles & Xie, 2023)). The output distribution parameters serve as the prior in the KL divergence between the encoder posterior and the dynamics prior:

$$\ell_t^k = \mathrm{KL}\Big(q_\phi(e_t^k \mid o_t) \,\|\, p_\xi(e_t^k \mid \{e_{<t}\}, a_{<t})\Big).$$

Intuitively, this KL term measures how surprising the current entity state is under the dynamics model's prediction. Because the model is entity-centric, it tends to assign low error to simple, familiar dynamics and higher error to states involving novel or complex interactions between entities. This makes the KL term a natural exploration bonus: states with poorly predicted entity transitions are precisely the ones that are likely to reveal new structure. We therefore define the intrinsic reward as the clipped per-entity KL divergence, averaged over all K entities:

$$\mathcal{L}_t^{\mathrm{dyn}} = \frac{1}{K}\sum_{k=1}^{K} \mathrm{clip}\big(\ell_t^k,\, 0,\, \mathrm{KL}_{\max}\big)\,. \tag{4}$$

Clipping prevents a single entity with unusually large prediction error from dominating the reward, since the KL is unbounded. Next, we discuss the online training of ECLAIR.

4.3 Online Training of ECLAIR with RL

A key challenge in pixel-based entity-centric exploration is that the learned entity representations evolve throughout training. As the policy drives the agent into previously unseen regions of the environment, it encounters configurations and object arrangements that were not present in the data used to initialize the representation model. This is particularly important in our setting, since unlike prior entity-centric RL work (Haramati et al., 2024; Qi et al., 2025), we do not assume that the pretraining data covers all relevant entities and interactions. If the encoder is kept frozen, it may fail to represent these novel configurations faithfully, which in turn produces stale and misleading exploration bonuses.

A second issue is that the intrinsic reward depends on the dynamics model's uncertainty, and this uncertainty is only informative when the model is updated on the same distribution induced by the current policy. If the dynamics model is trained only once and then fixed, its prediction error quickly becomes misaligned with the agent's evolving experience, weakening the connection between surprise and novelty. To avoid this drift, we jointly train the entity encoder, the dynamics model, and the RL policy online, so that both the representation and the intrinsic reward adapt as exploration progresses.

ECLAIR is agnostic to the choice of online RL algorithm. In our implementation, ECLAIR follows the setup of RND (Burda et al., 2018): we use PPO as the base RL algorithm and maintain separate value heads for extrinsic reward V^e and intrinsic reward V^i, with separate GAE returns for each. The combined advantage is $\hat{A}_t = \hat{A}_t^e + \beta \hat{A}_t^i$, where β scales the intrinsic exploration bonus. Rather than computing the intrinsic reward at every step, ECLAIR computes the KL-based bonus once every H steps as the average per-timestep prediction error accumulated over the window, and assigns it to the observation at the start of the window:

$$r_{t-H}^i = \frac{1}{H \cdot K}\sum_{h=0}^{H-1}\sum_{k=1}^{K} \mathrm{clip}\big(\ell_{t-H+h}^k,\ 0,\ \mathrm{KL}_{\max}\big)$$

(a) MINIHACK (Reward ↑)

Method	KeyRoom-F	KeyRoom-R	MultiRoom-N2	MultiRoom-N4	MultiRoom-N6	River
PPO (default)	0.24 ± 0.24	0.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{0.23 \pm 0.06}$
ε-greedy	$\mathbf{0.98 \pm 0.00}$	$\mathbf{0.99 \pm 0.00}$	$\mathbf{1.00 \pm 0.00}$	-0.10 ± 0.03	-0.04 ± 0.01	$\mathbf{0.21 \pm 0.07}$
Pseudocount	-0.02 ± 0.00	-0.26 ± 0.01	0.65 ± 0.03	-0.25 ± 0.00	-0.23 ± 0.01	0.09 ± 0.06
RND	0.00 ± 0.00	0.00 ± 0.00	$\mathbf{1.00 \pm 0.00}$	0.58 ± 0.05	-0.01 ± 0.01	$\mathbf{0.19 \pm 0.06}$
Dynamics	-0.04 ± 0.02	-0.18 ± 0.18	$\mathbf{1.00 \pm 0.00}$	-0.02 ± 0.01	-0.02 ± 0.01	0.14 ± 0.06
ECLAIR (ours)	$\mathbf{0.97 \pm 0.01}$	$\mathbf{0.98 \pm 0.01}$	0.99 ± 0.00	$\mathbf{0.99 \pm 0.00}$	$\mathbf{0.89 \pm 0.08}$	$\mathbf{0.21 \pm 0.04}$

(b) SPRITEWORLD (Norm. Reward ↑)

Method	DynamicHit	DynamicHit-H
PPO (default)	0.59 ± 0.09	0.57 ± 0.10
ε-greedy	0.32 ± 0.05	0.21 ± 0.02
Pseudocount	0.15 ± 0.02	0.25 ± 0.05
RND	0.49 ± 0.06	0.54 ± 0.04
ECLAIR (ours)	$\mathbf{0.74 \pm 0.08}$	$\mathbf{0.82 \pm 0.07}$

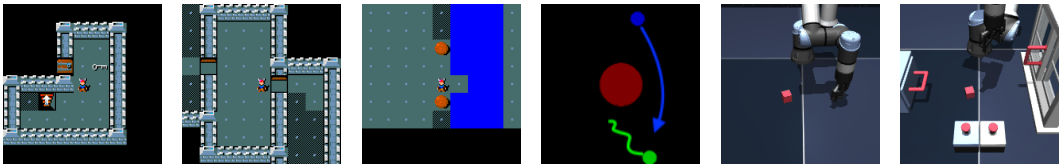
(c) OGBENCH (Success Rate ↑)

Method	Cube	Scene-T1	Scene-T2
PPO (default)	0.40 ± 0.06	0.48 ± 0.28	0.29 ± 0.09
ε-greedy	0.35 ± 0.05	0.00 ± 0.00	0.28 ± 0.04
Pseudocount	0.35 ± 0.05	$\mathbf{0.75 \pm 0.25}$	0.63 ± 0.18
RND	0.30 ± 0.04	$\mathbf{0.70 \pm 0.24}$	0.63 ± 0.18
Dynamics	0.48 ± 0.10	$\mathbf{0.85 \pm 0.10}$	0.63 ± 0.15
ECLAIR (ours)	$\mathbf{0.90 \pm 0.04}$	$\mathbf{0.90 \pm 0.04}$	$\mathbf{0.84 \pm 0.09}$

Table 1: Performance comparison of different exploration methods in visual domains. We report mean $\pm$ std. error over 4 seeds. We report reward for MINIHACK, normalized reward for SPRITEWORLD, and success rate for OGBENCH. **Bold** indicates results within one standard error of the best.

In practice, this significantly reduces training time by limiting how frequently the dynamics model is queried, and we find empirically that it does not hurt exploration quality. The full training procedure is summarized in Algorithm 1, where we highlight the modifications over PPO in blue. Additional implementation details on the dynamics model, KL computation, and hyperparameters of ECLAIR are provided in Appendix C.

5 Experiments



(a) KeyRoom (b) MultiRoom (c) River (d) DynamicHit (e) Cube (f) Scene

Figure 3: Representative environments from the three benchmarks considered in this work, spanning skill-based navigation, dynamic control, and robotic manipulation.

We design our experiments to answer three questions: **(1)** does ECLAIR improve over non-entity-centric exploration on hard-exploration visual RL tasks? **(2)** Does ECLAIR identify novel scenarios that facilitate long-horizon task solving? **(3)** How does each design choice affect ECLAIR's performance? To address these questions, we compare ECLAIR against several competitive exploration-based baselines on a diverse set of challenging visual RL tasks.

5.1 Long-Horizon Visual RL Tasks with Sparse Reward

We evaluate ECLAIR on visual RL tasks with sparse rewards across MiniHack (Samvelyan et al., 2021), SpriteWorld (Chuck et al., 2025), and OGBench (Park et al., 2025), benchmarking its exploration capabilities against strong baselines. To isolate the effect of exploration strategy, all methods are built on the same PPO (Schulman et al., 2017) implementation and differ solely in the exploration bonus applied during training.

Environments. We consider a total of 11 tasks spanning three benchmarks, all providing minimal reward signal until the agent completes the final goal. MiniHack focuses on skill-based

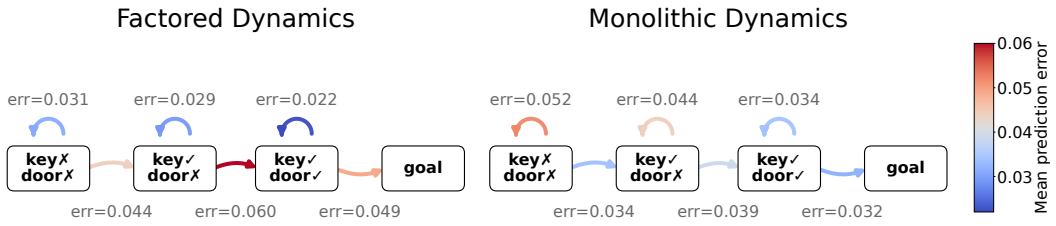


Figure 4: Case study on an abstract `KeyRoom` MDP comparing prediction errors from factored and monolithic dynamics models. Nodes denote task-progress states and edge colors show the mean held-out prediction error for each transition. The factored dynamics model assigns larger error (red) to task-critical transitions, such as acquiring the key and unlocking the door, whereas the monolithic dynamics model produces more uniform errors and does not clearly distinguish the transitions most relevant for solving the task.

navigation in procedurally generated grid-world environments, SpriteWorld tests exploration in dynamic control tasks where the agent must control a ball to bounce a target ball into a goal region, and OGBench evaluates long-horizon robotic manipulation. We evaluate on six MiniHack tasks, `KeyRoom-Fixed`, `KeyRoom-Random`, `MultiRoom-N2/N4/N6`, and `River`; two SpriteWorld tasks, `DynamicHit` and `DynamicHit-Hard`, and three OGBench tasks, `Cube`, `Scene-T1`, and `Scene-T2`.

All agents observe RGB observations without access to additional state information. We report task reward for MiniHack, success rate for OGBench, and normalized reward for SpriteWorld. All metrics have a maximum of 1.0. A visualization of the environments is provided in Figure 3, and full task implementation details are in Appendix D.

Baselines. We compare against several exploration methods with demonstrated effectiveness on image observations. *Pseudocount* (Bellemare et al., 2016) trains a density model to estimate state visitation counts. *RND* (Burda et al., 2018) uses random network distillation error as an exploration bonus. *Dynamics* follows Stadie et al. (2015), and computes latent prediction error bonuses. We adapt DreamerV3 (Hafner et al., 2023b) for our tasks, since the original model cannot learn any meaningful dynamics. We additionally compare against ε-greedy exploration and PPO with no explicit exploration signal as simple but competitive baselines. Implementation details are in Appendix E.

Results. As shown in Figure 6, ECLAIR matches or outperforms all baselines across every environment, demonstrating consistently strong exploration throughout. On the most challenging tasks: `MultiRoom-N4`, `MultiRoom-N6`, `Cube`, and `Scene-T2`, ECLAIR is the only method that reliably solves the task, while all baselines either fail entirely or degrade significantly. On easier tasks such as `KeyRoom-F`, `KeyRoom-R`, and `Scene-T1`, ECLAIR matches the best-performing baseline in each setting. The per-baseline failure modes are informative. ε-greedy excels in discrete action spaces, where random actions naturally yield meaningful exploratory behavior, but degrades sharply in continuous control settings. Prediction-based methods (RND, Dynamics) perform better in visually complex domains with continuous actions, consistent with their inductive bias toward modeling environment dynamics. Notably, all methods fail on `River`, the longest-horizon and partially observable task in our suite, suggesting that exploration bonuses alone are insufficient and that structured world knowledge or memory may be necessary for such settings.

The empirical finding that no prior exploration strategy dominates across all tasks is consistent with observations from a prior work (Taiga et al., 2020). We similarly find that baseline exploration methods exhibit domain-dependent strengths and weaknesses across our visual sparse-reward environments. ECLAIR proves to be the method that outperforms the baselines across our diverse benchmarks. Additional results, including training curves, can be found in Figure 6 in Appendix F.

5.2 Analysis on the Intrinsic Reward

A case study on `KeyRoom`. To illustrate why factored representations are useful for exploration, we analyze prediction errors in an abstract `KeyRoom` MDP using two dynamics models: an entity-centric model with a lightweight Transformer, which mirrors the structural bias used by ECLAIR, and a

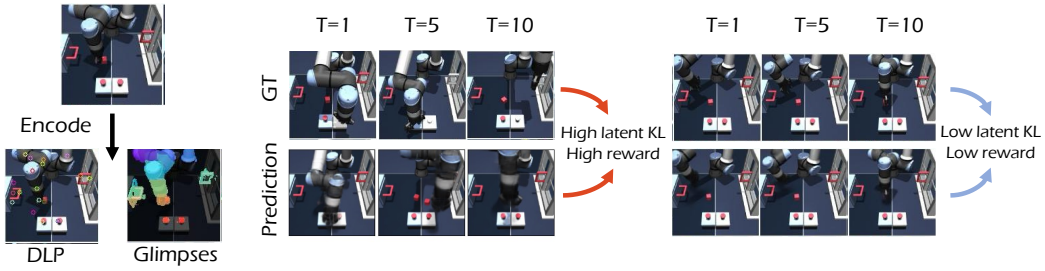


Figure 5: Visualization of ECLAIR's entity encoding and latent dynamics prediction on `Scene-T2` during RL training. When the predicted rollout differs from the ground truth (i.e. high latent KL divergence), the model rewards a large bonus, encouraging exploration of hard-to-predict entity states.

flat baseline that represents the full monolithic state and predicts dynamics with an MLP. We first collect an offline buffer of interaction data using a suboptimal agent, ensuring that the buffer contains coverage over all key interactions in `KeyRoom`. We then categorize the collected transitions into an abstract MDP over key interaction states: `(key` $\times$ `, door` $\times$ `)` $\rightarrow$ `(key`$\checkmark$`, door` $\times$ `)` $\rightarrow$ `(key`$\checkmark$`, door`$\checkmark$`)` $\rightarrow$ `goal`. Both dynamics models are trained on these categorized transitions and evaluated on a held-out set. As shown in Figure 4, the entity-centric dynamics model assigns substantially higher prediction error to transitions that require novel entity interactions, such as picking up the key and unlocking the door. In contrast, the monolithic MLP baseline distributes prediction error more uniformly across transitions, failing to distinguish which transitions are most informative. This suggests that entity-centric uncertainty provides a more targeted intrinsic reward signal, helping exploration focus on novel experiences for task completion.

Visual analysis on the intrinsic reward. Following the case study, we visualize how ECLAIR produces its intrinsic reward during RL training. As shown in Figure 5, ECLAIR encodes the image DLP latents and then uses its latent dynamics model to predict future entity states. When dynamics prediction differs substantially from the encoded rollout, ECLAIR returns a high latent KL value and thus a large intrinsic reward for these transitions, and vice versa. This illustrates that ECLAIR's exploration signal rewards meaningful object-state changes. We visualize the intrinsic rewards predicted by ECLAIR and RND along a trajectory in Figure 1 to show that ECLAIR produces a more meaningful exploration signal. More visualizations of ECLAIR's intrinsic rewards can be found in Appendix F.

5.3 Ablation Studies

We ablate the key design choices of **ECLAIR** on `Scene-T1`, reporting success rate averaged over 4 seeds. Results are summarized in Table 2, and training curves are in Figure 9 in the Appendix.

Representation. Replacing DLP with a patch-based encoder completely collapses performance, confirming that structured, object-level representations are essential for meaningful dynamics prediction. Removing online finetuning also leads to a notable performance drop, validating our design choice to continuously update the encoder as the policy explores new regions of the environment, since a frozen encoder fails to represent novel entity configurations.

Choice of Dynamics Loss. Replacing the latent KL-based bonus with pixel reconstruction error significantly degrades performance, suggesting that

Method	Success Rate
Representation	
w/o DLP (Patch-Based Encoder)	0.00 ± 0.00
w/o Online Finetune	0.61 ± 0.18
Exploration	
w/ Pixel Reconstruction Error	0.33 ± 0.20
w/o KL Clipping	0.63 ± 0.21
Hyperparameters	
Prediction Horizon $H = 1$	$\mathbf{0.92 \pm 0.03}$
Prediction Horizon $H = 5$	$\mathbf{0.90 \pm 0.04}$
Intrinsic Scale $\beta = 0.1$	0.55 ± 0.25
Intrinsic Scale $\beta = 0.5$	$\mathbf{0.87 \pm 0.03}$
Intrinsic Scale $\beta = 2.0$	0.00 ± 0.00
ECLAIR (full)	$\mathbf{0.90 \pm 0.04}$

Table 2: Ablation study on OGBENCH Scene-T1. Mean $\pm$ std error over 4 seeds (success rate $\uparrow$).

raw pixel errors are a less informative exploration signal than structured latent prediction errors. Removing per-entity clipping also hurts performance: without clipping, a single entity with anomalously high prediction error can dominate the bonus and destabilize training.

Hyperparameters. ECLAIR is robust to the prediction interval H, with $H \in \{1, 5, 10\}$ all achieving strong performance, suggesting that sparser bonus computation is viable without sacrificing exploration quality. The intrinsic reward scale β has more significant effect: too small ($\beta = 0.1$) provides an insufficient exploration signal, while too large ($\beta = 2.0$) overwhelms the extrinsic reward and causes complete task failure. Performance remains strong across $\beta \in \{0.5, 1.0\}$, indicating a reasonable range of valid values.

6 Limitations and Conclusion

While ECLAIR improves exploration across a diverse set of sparse-reward visual RL tasks, it has several limitations. First, learning and updating the factored representation and latent dynamics model introduces additional computational overhead. ECLAIR requires an initial warm-up phase before RL training, and computing the intrinsic reward requires querying the dynamics model during rollouts. Improving the efficiency of the model is an important direction for future work. Second, ECLAIR is less effective in environments with strong partial observability. The failure on `River` suggests that entity-centric novelty alone may be insufficient for tasks requiring long-term memory or structured world knowledge. Finally, ECLAIR treats the RL algorithm as a separate module and only modifies the exploration signal. Future work could explore tighter integration between ECLAIR and policy optimization, such as off-policy RL or model-based planning.

We introduced ECLAIR, an entity-centric exploration method for sparse-reward visual RL. ECLAIR learns factored representations online and uses latent dynamics prediction error as a structured intrinsic reward. We theoretically connected the dynamics prediction error to pseudo-count exploration and showed that factorizing novelty over entities yields can reduce effective exploration complexity compared to monolithic state-space exploration. Empirically, ECLAIR produces intrinsic rewards aligned with meaningful task-progress transitions and outperforms strong exploration baselines across 11 challenging visual RL tasks. These results demonstrate that entity-centric structure provides a powerful inductive bias for exploration in high-dimensional visual environments.

References

Peter Auer, Nicolo Cesa-Bianchi, and Paul Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine learning*, 47(2):235–256, 2002.

Marc Bellemare, Sriram Srinivasan, Georg Ostrovski, Tom Schaul, David Saxton, and Remi Munos. Unifying count-based exploration and intrinsic motivation. *Advances in neural information processing systems*, 29, 2016.

Yuri Burda, Harrison Edwards, Amos Storkey, and Oleg Klimov. Exploration by random network distillation. *arXiv preprint arXiv:1810.12894*, 2018.

Caleb Chuck, Supawit Chockchowwat, and Scott Niekum. Hypothesis-driven skill discovery for hierarchical deep reinforcement learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 5572–5579. IEEE, 2020.

Caleb Chuck, Kevin Black, Aditya Arjun, Yuke Zhu, and Scott Niekum. Granger causal interaction skill chains. *Transactions on Machine Learning Research*, 2024a. ISSN 2835-8856. URL https://openreview.net/forum?id=iA2KQyoun1.

Caleb Chuck, Sankaran Vaidyanathan, Stephen Giguere, Amy Zhang, David Jensen, and Scott Niekum. Automated discovery of functional actual causes in complex environments. *arXiv preprint arXiv:2404.10883*, 2024b.

Caleb Chuck, Fan Feng, Carl Qi, Chang Shi, Siddhant Agarwal, Amy Zhang, and Scott Niekum. Null counterfactual factor interactions for goal-conditioned reinforcement learning. *arXiv preprint arXiv:2505.03172*, 2025.

Tal Daniel and Aviv Tamar. Unsupervised image representation learning with deep latent particles. In *International Conference on Machine Learning*, pp. 4644–4665. PMLR, 2022.

Tal Daniel and Aviv Tamar. Ddlp: Unsupervised object-centric video prediction with deep dynamic latent particles. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=Wqn8zirthg.

Tal Daniel, Carl Qi, Dan Haramati, Amir Zadeh, Chuan Li, Aviv Tamar, Deepak Pathak, and David Held. Latent particle world models: Self-supervised object-centric stochastic dynamics modeling. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=lTaPtGiUUc.

Alejandro Escontrela, Ademi Adeniji, Wilson Yan, Ajay Jain, Xue Bin Peng, Ken Goldberg, Youngwoon Lee, Danijar Hafner, and Pieter Abbeel. Video prediction models as rewards for reinforcement learning. *Neural Information Processing Systems*, 2023.

Stefano Ferraro, Pietro Mazzaglia, Tim Verbelen, and Bart Dhoedt. Focus: object-centric world models for robotic manipulation. *Frontiers in Neurorobotics*, 19:1585386, 2025.

Daya Guo, Dejian Yang, Haowei Zhang, Junxiao Song, Peiyi Wang, Qihao Zhu, Runxin Xu, Ruoyu Zhang, Shirong Ma, Xiao Bi, et al. Deepseek-r1: Incentivizing reasoning capability in llms via reinforcement learning. *arXiv preprint arXiv:2501.12948*, 2025.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, 2020a.

Danijar Hafner, Timothy Lillicrap, Mohammad Norouzi, and Jimmy Ba. Mastering atari with discrete world models. *arXiv preprint arXiv:2010.02193*, 2020b.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023a.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse domains through world models. *arXiv preprint arXiv:2301.04104*, 2023b.

Dan Haramati, Tal Daniel, and Aviv Tamar. Entity-centric reinforcement learning for object manipulation from pixels. In *The Twelfth International Conference on Learning Representations*, 2024.

Dan Haramati, Carl Qi, Tal Daniel, Amy Zhang, Aviv Tamar, and George Konidaris. Hierarchical entity-centric reinforcement learning with factored subgoal diffusion. *arXiv preprint arXiv:2602.02722*, 2026.

Jiaheng Hu, Zizhao Wang, Peter Stone, and Roberto Martin-Martin. Elden: Exploration via local dependencies. *arXiv preprint arXiv:2310.08702*, 2023.

Shengyi Huang, Rousslan Fernand Julien Dossa, Chang Ye, Jeff Braga, Dipam Chakraborty, Kinal Mehta, and João G.M. Araújo. Cleanrl: High-quality single-file implementations of deep reinforcement learning algorithms. *Journal of Machine Learning Research*, 23(274):1–18, 2022. URL http://jmlr.org/papers/v23/21-1342.html.

Kevin Jamieson, Matthew Malloy, Robert Nowak, and Sébastien Bubeck. lil'ucb: An optimal exploration algorithm for multi-armed bandits. In *Conference on Learning Theory*, pp. 423–439. PMLR, 2014.

Diederik P. Kingma and Max Welling. Auto-encoding variational bayes. In Yoshua Bengio and Yann LeCun (eds.), *ICLR*, 2014.

Yitang Li, Zhengyi Luo, Tonghe Zhang, Cunxi Dai, Anssi Kanervisto, Andrea Tirinzoni, Haoyang Weng, Kris Kitani, Mateusz Guzek, Ahmed Touati, Alessandro Lazaric, Matteo Pirotta, and Guanya Shi. BFM-zero: A promptable behavioral foundation model for humanoid control using unsupervised reinforcement learning. In *The Fourteenth International Conference on Learning Representations*, 2026. URL https://openreview.net/forum?id=jkhl2oI0g5.

Sam Lobel, Akhil Bagaria, and George Konidaris. Flipping coins to estimate pseudocounts for exploration in reinforcement learning. In *International Conference on Machine Learning*, 2023.

Francesco Locatello, Dirk Weissenborn, Thomas Unterthiner, Aravindh Mahendran, Georg Heigold, Jakob Uszkoreit, Alexey Dosovitskiy, and Thomas Kipf. Object-centric learning with slot attention. *Advances in neural information processing systems*, 33:11525–11538, 2020.

Malte Mosbach, Jan Niklas Ewertz, Angel Villar-Corrales, and Sven Behnke. Sold: Slot object-centric latent dynamics models for relational manipulation learning from pixels. In *International Conference on Machine Learning*, pp. 44911–44935. PMLR, 2025.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations (ICLR)*, 2025.

Deepak Pathak, Pulkit Agrawal, Alexei A Efros, and Trevor Darrell. Curiosity-driven exploration by self-supervised prediction. In *International conference on machine learning*, pp. 2778–2787. PMLR, 2017.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF international conference on computer vision*, pp. 4195–4205, 2023.

Carl Qi, Dan Haramati, Tal Daniel, Aviv Tamar, and Amy Zhang. EC-diffuser: Multi-object manipulation via entity-centric behavior generation. In *The Thirteenth International Conference on Learning Representations*, 2025. URL https://openreview.net/forum?id=o3pJU5QCtv.

Benjamin Rolf, Ilya Jackson, Marcel Müller, Sebastian Lang, Tobias Reggelin, and Dmitry Ivanov. A review on reinforcement learning algorithms and applications in supply chain management. *International Journal of Production Research*, 61(20):7151–7179, 2023.

Mikayel Samvelyan, Robert Kirk, Vitaly Kurin, Jack Parker-Holder, Minqi Jiang, Eric Hambro, Fabio Petroni, Heinrich Kuttler, Edward Grefenstette, and Tim Rocktäschel. Minihack the planet: A sandbox for open-ended reinforcement learning research. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 1)*, 2021.

Cansu Sancaktar, Sebastian Blaes, and Georg Martius. Curious exploration via structured world models yields zero-shot object manipulation. *Advances in Neural Information Processing Systems*, 35:24170–24183, 2022.

John Schulman, Filip Wolski, Prafulla Dhariwal, Alec Radford, and Oleg Klimov. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Maximilian Seitzer, Bernhard Schölkopf, and Georg Martius. Causal influence detection for improving efficiency in reinforcement learning. *Advances in Neural Information Processing Systems*, 34: 22905–22918, 2021.

David Silver, Thomas Hubert, Julian Schrittwieser, Ioannis Antonoglou, Matthew Lai, Arthur Guez, Marc Lanctot, Laurent Sifre, Dharshan Kumaran, Thore Graepel, et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv preprint arXiv:1712.01815*, 2017.

Bradly C Stadie, Sergey Levine, and Pieter Abbeel. Incentivizing exploration in reinforcement learning with deep predictive models. *arXiv preprint arXiv:1507.00814*, 2015.

Adrien Ali Taiga, William Fedus, Marlos C. Machado, Aaron Courville, and Marc G. Bellemare. On bonus based exploration methods in the arcade learning environment. In *International Conference on Learning Representations*, 2020. URL https://openreview.net/forum?id=BJewlyStDr.

Zizhao Wang, Jiaheng Hu, Caleb Chuck, Stephen Chen, Roberto Martín-Martín, Amy Zhang, Scott Niekum, and Peter Stone. Skild: Unsupervised skill discovery guided by factor interactions. *Advances in Neural Information Processing Systems*, 37:77748–77776, 2024.

Andrii Zadaianchuk, Maximilian Seitzer, and Georg Martius. Self-supervised visual reinforcement learning with object-centric representations. In *International Conference on Learning Representations*, 2021.

Weipu Zhang, Adam Jelley, Trevor McInroe, and Amos Storkey. Objects matter: object-centric world models improve reinforcement learning in visually complex environments. *arXiv preprint arXiv:2501.16443*, 2025.

A Proofs and Extended Theoretical Results

A.1 Proof of Proposition 1

We instantiate the pseudo-count framework of Bellemare et al. (2016) in the latent space $\mathcal{Z}$, identifying the density model with the dynamics prior $\rho_n(z') := p_\xi(z' \mid z, a)$ and the recoding probability with $\rho'_n(z') := p_{\xi'_n}(z' \mid z, a)$, where ξ'_n are parameters after a single step of size α toward $q_\phi(z' \mid s')$, so that $\mu_{\xi'_n} = \mu_\xi + \alpha(\mu_\phi(s') - \mu_\xi(z, a))$. The learning-positivity condition $\rho'_n(z') \geq \rho_n(z')$ holds since the update increases the log-likelihood of $z' \sim q_\phi$.

Under Assumption 1, the information gain (IG) is the KL divergence between the two Gaussians. For approximately equal variances ($\sigma_\phi^2 \approx \sigma_\xi^2$), this reduces to $\mathrm{IG}_n(s') \approx \|\mu_\phi(s') - \mu_\xi(z, a)\|^2 / 2\sigma_\xi^2$. For the prediction gain, substituting the Gaussian log-densities and taking the expectation over $z' \sim q_\phi(z' \mid s')$ gives

$$\overline{\mathrm{PG}}_n(s') = \mathbb{E}_{z' \sim q_\phi}\left[\log \frac{\rho'_n(z')}{\rho_n(z')}\right] = \frac{\alpha(2-\alpha)\|\mu_\phi(s') - \mu_\xi(z, a)\|^2}{2\sigma_\xi^2} \approx 2\alpha \cdot \mathrm{IG}_n(s'),$$

where the last step uses $\alpha \ll 1$. Thus the prediction gain and information gain are proportional, both driven by $\|\mu_\phi(s') - \mu_\xi(z, a)\|^2$.

By Theorem 1 of Bellemare et al. (2016), $\mathrm{IG}_n(s') \leq \overline{\mathrm{PG}}_n(s') \leq \hat{N}_n(s')^{-1}$. Using the approximation $\hat{N}_n \approx (e^{\mathrm{PG}_n} - 1)^{-1} \approx \mathrm{PG}_n^{-1}$ for small PG, we obtain

$$\hat{N}_n(s') \approx \frac{\sigma_\xi^2}{\alpha\|\mu_\phi(s') - \mu_\xi(z, a)\|^2},$$

confirming that large prediction error yields small pseudo-count, i.e. high novelty. The intrinsic reward then satisfies

$$r^i(s, a) = \hat{N}_n(s')^{-1/2} \propto \frac{\|\mu_\phi(s') - \mu_\xi(z, a)\|}{\sigma_\xi}.$$

Finally, since $\mathrm{KL}(q_\phi(z' \mid s') \| p_\xi(z' \mid z, a)) = \mathrm{IG}_n(s') \propto \|\mu_\phi(s') - \mu_\xi(z, a)\|^2 / \sigma_\xi^2$, using the KL divergence directly as an exploration bonus is proportional to the same normalized prediction error, justifying its use in place of $\hat{N}_n(s')^{-1/2}$. □

A.2 Proof of Proposition 2

Under Assumption 2, the joint bandit problem decomposes into KE independent sub-problems, one for each entity-state pair $(k, e) \in [K] \times \mathcal{E}_k$. Each sub-problem becomes: given that entity k is in state e, finding the action $a^*(k, e) = \arg\max_a r_k(e, a)$ that maximizes the per-entity reward. This is an A-armed bandit with arm gaps $\Delta_k(e, a) = r_k(e, a^*(k, e)) - r_k(e, a)$. Crucially, each transition visiting joint state $s = (e^1, \ldots, e^K)$ under action a will give a noisy observation of $r_k(e^k, a)$ for every k simultaneously, so a single interaction increments the sample counter $n_k(e^k, a)$ for all K entity-state sub-problems at once.

Running lil' UCB (Jamieson et al., 2014) on each of the KE sub-problems at confidence $\delta' = \delta/(KE)$, Theorem 2 in Jamieson et al. (2014) gives an expected sample cost of $\sum_{a:\Delta_k(e,a)>0} \Theta(\Delta_k(e, a)^{-2} \log\log \Delta_k(e, a)^{-2})$ per sub-problem, with each returning the correct best arm with probability at least $1 - \delta'$. To ensure that all KE sub-problems return the correct best arm, we apply a union bound and argue that

$$P(\text{that any single sub-problem fails}) \leq KE \cdot \delta' = \delta$$

Therefore, the joint success probability is at least $1 - \delta$. Summing the sub-problem costs over all $(k, e) \in [K] \times \mathcal{E}_k$ gives

$$N_{\text{fact}} = \sum_{k=1}^{K} \sum_{e \in \mathcal{E}_k} \sum_{a:\, \Delta_k(e,a)>0} \Theta\big(\Delta_k(e, a)^{-2} \log\log \Delta_k(e, a)^{-2}\big)$$

□

B Extended Background

Count-based Exploration. Methods such as Auer et al. (2002); Bellemare et al. (2016) assign an exploration bonus inversely proportional to the visitation count of a state: $r_t^i = 1/\sqrt{N_t(s_t)}$, where $N_t(s_t)$ is the number of times state s_t has been visited up to time t. This originates from the Upper Confidence Bound (UCB) method from the bandit setting: for an arm pulled n times with empirical mean $\hat{\mu}$, the UCB index $\hat{\mu} + \Theta(1/\sqrt{n})$ ensures that arms with high uncertainty are explored preferentially. The lil' UCB algorithm (Jamieson et al., 2014) refines this using the law of the iterated logarithm and bound the samples required to identify the best among N arms to be $\Theta(\sum_{i \neq i^*} \Delta_i^{-2} \log\log \Delta_i^{-2})$, where $\Delta_i = \mu^* - \mu_i$ is the sub-optimality gap of arm i. This means that the total sample cost of UCB exploration is the *sum of per-arm costs* $\Delta_i^{-2} \log\log \Delta_i^{-2}$.

Bellemare et al. (2016) generalize this idea to high-dimensional spaces by replacing $N_t(s_t)$ with a pseudo-count $\hat{N}_t(s_t)$ derived from a learned density model over the state space. Concretely, given the density $\rho_n(s)$ before observing s, and the *recoding probability* $\rho'_n(s)$, which is the density assigned to s after after observing a new occurrence of s, the pseudo-count is defined as:

$$\hat{N}_n(s) = \frac{\rho_n(s)(1 - \rho'_n(s))}{\rho'_n(s) - \rho_n(s)}.$$

Intuitively, $\hat{N}_n(s)$ is small when the density model updates significantly upon seeing s, indicating that s is novel, and large when the model is already confident about s. The bonus $r^i = 1/\sqrt{\hat{N}_n(s)}$ then naturally decays as the agent accumulates experience.

Prediction-error-based Exploration. Stadie et al. (2015) propose using the prediction error of a learned dynamics model as an exploration bonus, motivating it as a proxy for state novelty: $r_t^i = \|f_\phi(s_t, a_t) - s_{t+1}\|^2$. Random Network Distillation (RND) (Burda et al., 2018) offers a simpler alternative, defining the bonus as the prediction error of a trainable network f_θ against a fixed randomly initialized target network f^*: $r_t^i = \|f_\theta(s_t) - f^*(s_t)\|^2$, which naturally decays for frequently visited states as f_θ improves its predictions. Intuitively, prediction errors are large for novel states and diminish with repeated visits, mirroring the pseudo-count bonus of Bellemare et al. (2016), though prior works treat this as an informal analogy rather than a formal connection. In the context of interactions, several works have explored utilizing prediction error for skill learning in hierarchical reinforcement learning Chuck et al. (2020; 2024a); Wang et al. (2024), which is motivated by a counterfactual description of interaction (Chuck et al., 2024b).

Latent Particle World Model (LPWM (Daniel et al., 2026)). LPWM (Daniel et al., 2026) is a self-supervised variational autoencoder (VAE (Kingma & Welling, 2014)) for end-to-end object-centric world modeling from pixels. The input images are encoded as a set of low-dimensional deep latent particles (DLP (Daniel & Tamar, 2022; 2024) and the autoregressive transformer-based dynamics module predicts the next step in the latent particle space, optionally conditioned on external signals such as actions, language or goal images.

Formally, DLP decomposes an image into a set of M latent foreground particles $\{z^i\}_{i=0}^{M-1}$ and a latent background particle z_{bg}, and is trained in an unsupervised manner as a VAE, where the objective is reconstructing the original input image from the latent particles. Each foreground particle consists of multiple attributes $z^i = [z_p, z_s, z_d, z_t, z_f]^i \in \mathbb{R}^{(6+n)}$: $z_p \in \mathbb{R}^2$ represents the position in 2D coordinates (i.e., a keypoint); $z_s \in \mathbb{R}^2$ denotes the scale, specifying the height and width of a bounding box around the particle; $z_d \in \mathbb{R}$ approximates local depth when particles are close together; $z_t \in \mathbb{R}$ is a transparency parameter in the range $[0, 1]$; and $z_f \in \mathbb{R}^n$ encodes visual features from a glimpse around the particle, where n is a hyper-parameter determining the latent dimension of the visual features. The background is encoded as $z_{\text{bg}} \in \mathbb{R}^{n_{\text{bg}}}$, with n_{bg} serving as the latent dimension hyper-parameter for the background encoding. LPWM extends the DLP framework to

Algorithm 1 ECLAIR: Entity-Centric Latent dynAmics Intrinsic Reward

Require: Policy π_θ, value heads V^e_ψ, V^i_ψ, LPWM encoder q_ϕ, dynamics prior p_ξ
Require: Intrinsic reward interval H, PPO epochs E_{ppo}, LPWM update steps N_{dyn}, intrinsic reward scale β, and $\text{KL}_{\max}$.

Let $\ell^k_t = \text{clip}\big(\text{KL}\big(q_\phi(e^k_t \mid o_t) \,\|\, p_\xi(e^k_t \mid \{e_{<t}\}, a_{<t})\big),\ 0,\ \text{KL}_{\max}\big)$

Warmup: Initialize replay buffer $\mathcal{R}$; collect data under random policy; pretrain (q_ϕ, p_ξ) on $\mathcal{R}$
for iteration $= 1, 2, \ldots$ **do**
  Initialize $r^i_t \leftarrow 0$ for all t
  for $t = 1, \ldots, T$ **do** ▷ Rollout
    $\{e^k_t\}_{k=1}^K \leftarrow q_\phi(o_t)$; sample $a_t \sim \pi_\theta(\cdot \mid \{e^k_t\})$; observe o_{t+1}, r^e_t
    if $t \mod H = 0$ **then** ▷ Compute bonus every H steps
      $$r^i_{t-H} \leftarrow \frac{1}{H \cdot K} \sum_{h=0}^{H-1} \sum_{k=1}^{K} \ell^k_{t-H+h}$$
    end if
    Store (o_t, a_t, r^e_t, r^i_t) in $\mathcal{B}$; store (o_t, a_t) in $\mathcal{R}$
  end for
  Compute $\hat{A}^e, \hat{A}^i$ via GAE using V^e_ψ, V^i_ψ; set $\hat{A}_t \leftarrow \hat{A}^e_t + \beta \hat{A}^i_t$
  for epoch $= 1, \ldots, E_{\text{ppo}}$; each minibatch $\mathcal{M} \subset \mathcal{B}$ **do**
    Update (θ, ψ) with PPO clipped objective on $\hat{A}_t$
  end for
  Sample sequences from $\mathcal{R}$; update (q_ϕ, p_ξ) for N_{dyn} steps ▷ LPWM update
end for

world modeling by introducing dynamics modules that operate directly in the low-dimensional latent space, predicting the next-step particles, and are trained end-to-end with the encoder and decoder.

C Method Details

The full training procedure of ECLAIR is summarized in Algorithm 1, where we highlight the modifications over PPO in blue.

LPWM architecture details. As introduced above, ECLAIR uses LPWM (Daniel et al., 2026) as its entity-centric latent dynamics model. We use the full LPWM particle set,

$$z^t = \{z^1_t, \ldots, z^M_t, z^t_{\text{bg}}\},$$

which includes M foreground particles and one background particle. ECLAIR computes prediction error over all particles, including the background particle, and uses this error as the intrinsic exploration signal. LPWM defines an autoregressive prior over the next-step particle set,

$$p_\xi(z^t \mid z^{0:t-1}, a^{0:t-1}),$$

where the conditioning includes the history of encoded particles and environment actions. The dynamics module takes the spatio-temporal particle sequence

$$\big\{\{z^m_\tau\}_{m=1}^M, z^\tau_{\text{bg}}\big\}_{\tau=0}^{t-1}$$

as input, together with the action history $\{a_\tau\}_{\tau=0}^{t-1}$. It is implemented as a causal spatio-temporal Transformer over particle tokens. Each block alternates between spatial attention over particles within the same timestep and temporal attention over each particle identity across time. The spatial block captures interactions between particles, while the temporal block captures particle-specific dynamics. Causal masking ensures that the prediction at time t depends only on particles and actions from previous timesteps.

Actions are incorporated by embedding the global environment action a_t, broadcasting it to all particle tokens at timestep t, and injecting it through adaptive layer normalization (AdaLN (Peebles & Xie, 2023)). The Transformer output is then mapped to the parameters of the predictive distributions for each particle attribute. Thus, the learned dynamics prior has the same attribute-wise factorization as the encoder posterior, allowing the KL to be computed separately for each foreground attribute and for the background particle. We refer the readers to the original LPWM paper for extensive details.

KL-divergence Computation. ECLAIR uses the same attribute-wise masked KL formulation as LPWM. In the standard VAE, each posterior attribute distribution is compared against a fixed prior. In the dynamics KL used by ECLAIR, the posterior at time t is instead compared against the learned autoregressive dynamics prior predicted from previous particles and actions.

For each foreground particle m, the per-particle dynamics KL is

$$\begin{aligned}\ell_t^m = &\sum_{\mathrm{att}\in\{o,s,d\}} \mathrm{KL}\big[q_\phi(z^m_{\mathrm{att},t} \mid o_t, z^m_{a,t}) \,\|\, p_\xi(z^m_{\mathrm{att},t} \mid z^{0:t-1}, a^{0:t-1})\big] \odot z^m_{t,t} \\ &+ \mathrm{KL}\big[q_\phi(z^m_{t,t} \mid o_t, z^m_{a,t}) \,\|\, p_\xi(z^m_{t,t} \mid z^{0:t-1}, a^{0:t-1})\big] \\ &+ \beta_f \mathrm{KL}\big[q_\phi(z^m_{f,t} \mid o_t, z^m_{p,t}) \,\|\, p_\xi(z^m_{f,t} \mid z^{0:t-1}, a^{0:t-1})\big] \odot z^m_{t,t}.\end{aligned}$$

Here, $\mathrm{att} \in \{o, s, d\}$ denotes offset, scale, and depth. Following LPWM, these explicit-attribute KL terms and the visual-feature KL are masked by the particle transparency $z^m_{t,t}$, so inactive particles do not contribute to them. The transparency KL itself is not masked, since the model must also predict whether particles become active or inactive.

The background particle contains only visual features, so its dynamics KL is

$$\ell_t^{\mathrm{bg}} = \beta_f \mathrm{KL}\big[q_\phi(z^t_{\mathrm{bg}} \mid o_t, z_{p,t}) \,\|\, p_\xi(z^t_{\mathrm{bg}} \mid z^{0:t-1}, a^{0:t-1})\big]\,.$$

The intrinsic reward is the clipped average over all foreground particles and the background particle:

$$r_t^i = \frac{1}{M+1}\left(\sum_{m=1}^{M} \mathrm{clip}(\ell_t^m, 0, \mathrm{KL}_{\max}) + \mathrm{clip}\Big(\ell_t^{\mathrm{bg}}, 0, \mathrm{KL}_{\max}\Big)\right).$$

Clipping prevents a single particle with unusually large prediction error from dominating the reward, while averaging keeps the reward scale stable. Since the KL is computed at the particle and attribute level, the resulting reward highlights poorly predicted object interactions, particle activation changes, and global scene changes captured by the background particle.

Training details. We first pre-train the LPWM model on randomly collected interaction data to warm up both the particle representation and the latent dynamics prior. During online RL training, trajectories collected by the current policy are added to a replay buffer, which is used to continually update LPWM. The buffer is maintained with a replacement strategy so that newly collected on-policy rollouts gradually replace older random experiences, allowing the dynamics model to adapt to the state distribution induced by the improving policy.

In practice, we find that LPWM requires a relatively high update-to-data ratio during online training. As the policy discovers new states and interactions, the particle encoder and dynamics prior must quickly adapt to the changing replay distribution; otherwise, the intrinsic reward can reflect stale representation error rather than meaningful dynamics prediction error. We therefore update LPWM for multiple gradient steps per batch of newly collected rollouts so that the representation and dynamics model can catch up to the agent's most recent experiences.

We also experimented with maintaining an exponential moving average (EMA) of the LPWM parameters to stabilize the intrinsic reward computation. However, we did not observe a consistent benefit from EMA in our experiments, and therefore use the current LPWM parameters directly when computing the dynamics KL and the corresponding intrinsic reward.

Hyperparameter	MiniHack	SpriteWorld	OGBench
Total environment steps	500K–5M	3M–5M	5M
Parallel environments	32	32	32
Rollout length per env.	128	128	512
PPO batch size	32×128	32×128	32×512
Learning rate	1e−4	1e−4	3e−4
Discount γ	0.99	0.99	0.99
GAE parameter λ	0.95	0.95	0.95
PPO update epochs E_{ppo}	4	4	10
Number of minibatches	4	4	8
Clip coefficient	0.1	0.1	0.2
Clipped value loss	Yes	Yes	Yes
Entropy coefficient	0.01	0.01	0.0
Value loss coefficient	0.5	0.5	1.0
Maximum gradient norm	0.5	0.5	0.5
Target KL	None	None	0.01
Intrinsic reward interval H	10	10	10
Intrinsic scale β	1.0	1.0	1.0
LPWM update steps N_{dyn}	5	30	30
LPWM warmup collection steps	10K	7K	50K
LPWM warmup epochs	10	20	15
LPWM batch size	32	16	16
LPWM learning rate	1e−4	8e−5	8e−5

Table 3: Domain-specific hyperparameters for ECLAIR.

We normalize the intrinsic reward online using a running mean and standard deviation computed from the stream of intrinsic bonuses observed during training. Specifically, before combining the intrinsic reward with the extrinsic reward, we update a running reward normalizer and scale the intrinsic bonus by the running standard deviation. This prevents the magnitude of the exploration bonus from drifting as LPWM improves or as the policy visits new regions of the state space, and makes the intrinsic-reward scale comparable across training stages.

Hyperparameters. We provide ECLAIR's hypeparameters for each specific domain in Table 3. We tune rollout length, episode horizon, LPWM warmup, and LPWM update-to-data ratio by domain, since the representation and dynamics model must adapt at different rates across different domains.

Hardware and computational resource requirements. For ECLAIR training, we use NVIDIA A40 GPUs with 40GB of memory each. SpriteWorld experiments use 2 GPUs, while MiniHack and OGBench experiments use 4 GPUs. PPO training is run on GPU 0, and the remaining GPUs are used to train the representation and dynamics models. Each experiment with 5M environment steps takes approximately 12–15 hours to complete.

D Environment Details

We consider a total of 11 tasks spanning three benchmarks, MiniHack, SpriteWorld, and OGBench. All environments emit 128×128 image observations and provide minimal reward signal until the agent completes the final goal. We list the tasks and provide detailed descriptions for each below:

`KeyRoom-Fixed`: agent picks up a key and unlocks a door at fixed locations to reach the goal.

`KeyRoom-Random`: same as above, but with randomized entity locations each episode.

`MultiRoom-N2/N4/N6`: agent navigates through N rooms with doors, with difficulty scaling with N.

`River`: agent pushes boulders into water to create a traversable path across the river.

`DynamicHit`: agent controls a ball and bounces a target ball into a goal region.

`DynamicHit-Hard`: a harder variant of `DynamicHit` with a smaller balls and goal region.

`Cube`: robot transports a cube to a fixed target location.

`Scene-T1`: robot opens both the drawer and the window.

`Scene-T2`: robot first unlocks and then opens both the drawer and the window.

The MiniHack (Samvelyan et al., 2021) domain. MiniHack (released under the Apache-2.0 License) contains grid-based environments built on top of NetHack, where the agent observes a rendered partial game state and interacts with discrete objects such as keys, doors, boulders, water, and staircases. The tasks require temporally extended exploration because reward is **only** obtained after completing the full interaction sequence. Across all MiniHack tasks, we use a shared discrete action space of 10 actions, including movement and NetHack interaction actions such as pickup and apply. Many actions are task-irrelevant and have no effect in some environments, further increasing the exploration challenge.

In `KeyRoom`, the agent starts in an outer room containing a key and must first navigate to the key, pick it up, move to a locked door, apply the key to unlock the door, and finally reach the staircase inside the locked room. We evaluate two variants: `KeyRoom-Fixed`, where the key, door, and staircase locations are fixed, and `KeyRoom-Random`, where these locations are randomized across episodes. This task tests whether an exploration method can discover and reuse a sparse sequence of object interactions: key acquisition, door unlocking, and goal reaching.

The `MultiRoom` tasks are procedurally generated navigation tasks consisting of a chain of connected rooms. The agent must move through the rooms in sequence, open or pass through the doors connecting adjacent rooms, and reach the final goal location in the last room. We evaluate `MultiRoom-N2`, `MultiRoom-N4`, and `MultiRoom-N6`, where the number denotes the number of rooms. Increasing the number of rooms lengthens the required exploration horizon and increases the amount of irrelevant navigation before the agent receives any successful terminal reward. These tasks therefore test long-horizon sparse-reward exploration under increasingly large procedurally generated layouts.

The `River` task requires both navigation and object manipulation. The agent starts on one side of a river and must reach the goal on the opposite side. Water tiles are not directly traversable, but boulders can be pushed into the water to create dry land tiles. Solving the task therefore requires the agent to discover that boulders are useful objects, push them into appropriate water locations, and construct a connected path across the river before navigating to the goal. Unlike simple navigation tasks, `River` requires the agent to perform a sequence of environment-altering interactions before the goal becomes reachable.

The SpriteWorld (Chuck et al., 2025) domain. `DynamicHit` and `DynamicHit-Hard` (released under the Apache-2.0 License) involve controlling a 2D "control ball" using force control in a 10m $\times$ 10m field bounded by walls, with coordinates in the center. The objective is to use the control ball to manipulate a second 2D ball of the same size into the goal area, which is a 1.3m diameter circle in the center of the field. The starting position of the control ball is always initialized to the center-bottom at position $(0\text{m}, -2\text{m})$, while the target ball can be initialized anywhere outside the goal with zero velocity. The state space is a 128×128 pixel rendering of the scene, with the control ball rendered in blue and the target ball in green, and the goal rendered in red. The reward function is $+1$ at the goal and 0 elsewhere. Finally, the distinction between `DynamicHit` and `DynamicHit-Hard` is in the size of the ball, which has a radius of 0.5m in `DynamicHit`, and 0.4m in `DynamicHit-Hard`.

The OGBench (Park et al., 2025) domain. OGBench (released under the MIT License) consists of continuous-control manipulation tasks with a UR5e arm and a Robotiq 2F-85 gripper. The agent observes third-person RGB renderings from MuJoCo and outputs a 5-D action corresponding to relative end-effector translation, yaw rotation, and gripper opening. We adapt the environment from Haramati et al. (2026) and remove goal conditioning to study the single-task online RL setting.

In `Cube`, the arm must transport a cube to a fixed target location at $(0.3, -0.2)$ on the table. The cube is initialized at a random start position elsewhere in the workspace at the beginning of each episode. Solving the task requires the agent to reach the cube, grasp it with the gripper, lift it, move it to the target, and place it down. The reward is sparse: the agent receives a positive reward only when the cube is within tolerance of the target location. Thus, the exploration method must discover the full reach–grasp–transport–place sequence without intermediate supervision.

The `Scene-T1` and `Scene-T2` tasks use a tabletop scene containing a drawer, a sliding window, and two buttons that control the locks for the drawer and window, respectively. In `Scene-T1`, both locks are initially unlocked, and the agent must open both the drawer, by pulling its handle outward, and the window, by sliding its handle sideways, to reach the goal configuration. In `Scene-T2`, both locks are initially locked, so the agent must first press button 0 to unlock the drawer and button 1 to unlock the window before opening them. This creates a longer compositional task with a sequential dependency between unlocking and opening each object. To make these long-horizon manipulation tasks tractable while retaining sparse feedback, we use stepwise sparse rewards for `Scene-T1` and `Scene-T2`: the agent receives sparse rewards upon completing key subgoals, such as unlocking an object or opening it, rather than only after the entire scene reaches the final goal configuration.

Evaluation protocol. For evaluation, we report the average performance of each method over the last 5 evaluation checkpoints of training, corresponding to 5×32 trajectories per seed. We report the mean and standard err across 4 seeds in our environment tables.

E Baseline Implementation Details

All methods, including ECLAIR and the baselines, are implemented on top of PPO in CleanRL (Huang et al., 2022) [1]. We use the same training backbone whenever possible to ensure a controlled comparison. Unless otherwise stated, each method uses the same PPO optimizer, rollout collection setup, actor-critic architecture, environment wrappers, and reward normalization pipeline, differing only in the exploration mechanism used to produce intrinsic rewards or exploratory actions.

PPO. We implement the no-exploration baseline as the CleanRL PPO backbone trained only with the environment reward. For all continuous-control environments, the policy outputs a diagonal Gaussian action distribution and is optimized with parallel environments, learning rate with linear annealing, discount $\gamma = 0.99$, GAE parameter $\lambda = 0.95$. The actor-critic uses the same CNN encoder as the exploration baselines. All domain-specific PPO hyperparameters are provided in Table 3.

ε-greedy. The ε-greedy baseline uses the same PPO training objective and network architecture as the no-exploration PPO baseline, but injects random actions during data collection. Since our OGBench environments use continuous actions, with probability ε we replace the policy action with a uniformly sampled action from the environment action space; otherwise, we execute the sampled PPO action. We linearly anneal ε from 1.0 to 0.05 over the first 100K environment steps, and then keep it fixed at 0.05 for the remainder of training.

Pseudocount. The pseudocount baseline follows the intrinsic-motivation formulation of Bellemare et al. (2016). We train a density model over image observations and use the change in assigned probability before and after updating the model on the current observation to derive a pseudocount estimate. The intrinsic reward is then computed as a count-based exploration bonus from this pseudocount, encouraging the agent to visit states that are unlikely under the learned density model.

[1] `https://github.com/vwxyzjn/cleanrl`. CleanRL is released under the MIT License.

As with the other intrinsic-reward baselines, we normalize the resulting bonus online and combine it with the extrinsic reward in PPO using separate intrinsic and extrinsic value heads.

RND. For RND, we follow the standard random network distillation setup (Burda et al., 2018). A fixed randomly initialized target network and a trainable predictor network both take image observations as input, and the squared prediction error of the predictor is used as the intrinsic reward. We initialize observation normalization using 50 iterations of random rollouts, train the predictor on 25% of collected samples per update, and normalize the intrinsic reward online before combining it with the extrinsic PPO advantage. PPO uses separate value heads for extrinsic and intrinsic returns.

Dynamics. The Dynamics baseline follows the prediction-error exploration principle of Stadie et al. (2015). We train an action-conditioned latent dynamics model from online interaction data and use its prediction error as the intrinsic reward. Concretely, the model predicts future latent observations conditioned on recent observations and actions, and the intrinsic bonus is computed from the discrepancy between the predicted and observed future representations. The dynamics model is updated online from a replay buffer collected during PPO training, while the policy is updated on on-policy PPO rollouts. As with RND, the intrinsic reward is normalized and combined with the extrinsic reward using separate intrinsic and extrinsic value heads.

DreamerV3. We adapt DreamerV3 (Hafner et al., 2023a) to the Dynamics baseline. The RSSM observes raw image observations and is modified to support the action space of each domain. We use the model's latent prediction error, computed as the KL between posterior and prior latent states, as the intrinsic reward for PPO. For a fair comparison, DreamerV3 receives the same interaction data and training budget as ECLAIR, including the same warmup samples, online samples, warmup training budget, and online update budget. Our DreamerV3 curiosity baseline is comparable in scale to the implementation from `dreamerv3-torch` [2] . In MiniHack, we increase the deterministic RSSM state to 3072 to match the provided MiniHack configuration, resulting in a substantially larger dynamics model. The resulting curiosity signal is normalized and combined with the extrinsic reward using the same PPO intrinsic-reward pipeline as the other exploration baselines.

F Additional Results

ECLAIR intrinsic reward visualizations. To better understand the exploration signal learned by ECLAIR, we visualize the intrinsic reward along representative rollouts in Figures 7 and 8. Figure 7 shows a trajectory segment from `KeyRoom`, together with the corresponding predictions from the entity-centric dynamics model. We observe that the intrinsic reward peaks when the agent performs a meaningful environment-changing interaction: opening the locked door after acquiring and applying the key. This transition is difficult to predict and directly relevant to task progress, since it changes the set of reachable states and enables the agent to proceed toward the goal.

We observe a similar pattern in the continuous-control manipulation domain. In Figure 8, the intrinsic reward spikes when the agent opens the drawer in `Scene-T1`. This interaction changes the scene configuration and corresponds to one of the key subgoals required for task completion. Together, these visualizations suggest that ECLAIR does not merely reward visually novel states, but instead assigns high bonus to informative object interactions that alter the environment dynamics.

Training curves. Figure 6 shows the full training curves across all 11 environments. Overall, ECLAIR achieves the strongest final performance on most tasks and is particularly robust on the harder long-horizon environments. In SpriteWorld, ECLAIR improves steadily throughout training and outperforms baselines on both `DynamicHit` and `DynamicHit-Hard`, indicating that entity-centric prediction provides a useful exploration signal in continuous object-interaction domains. In MiniHack, ECLAIR is competitive on the easier `KeyRoom-Fixed`, `KeyRoom-Random`, and `MultiRoom-N2` tasks, while showing a larger advantage on the harder `MultiRoom-N4` and `MultiRoom-N6` tasks, where most baselines plateau or fail to make consistent progress. On

[2] https://github.com/NM512/dreamerv3-torch

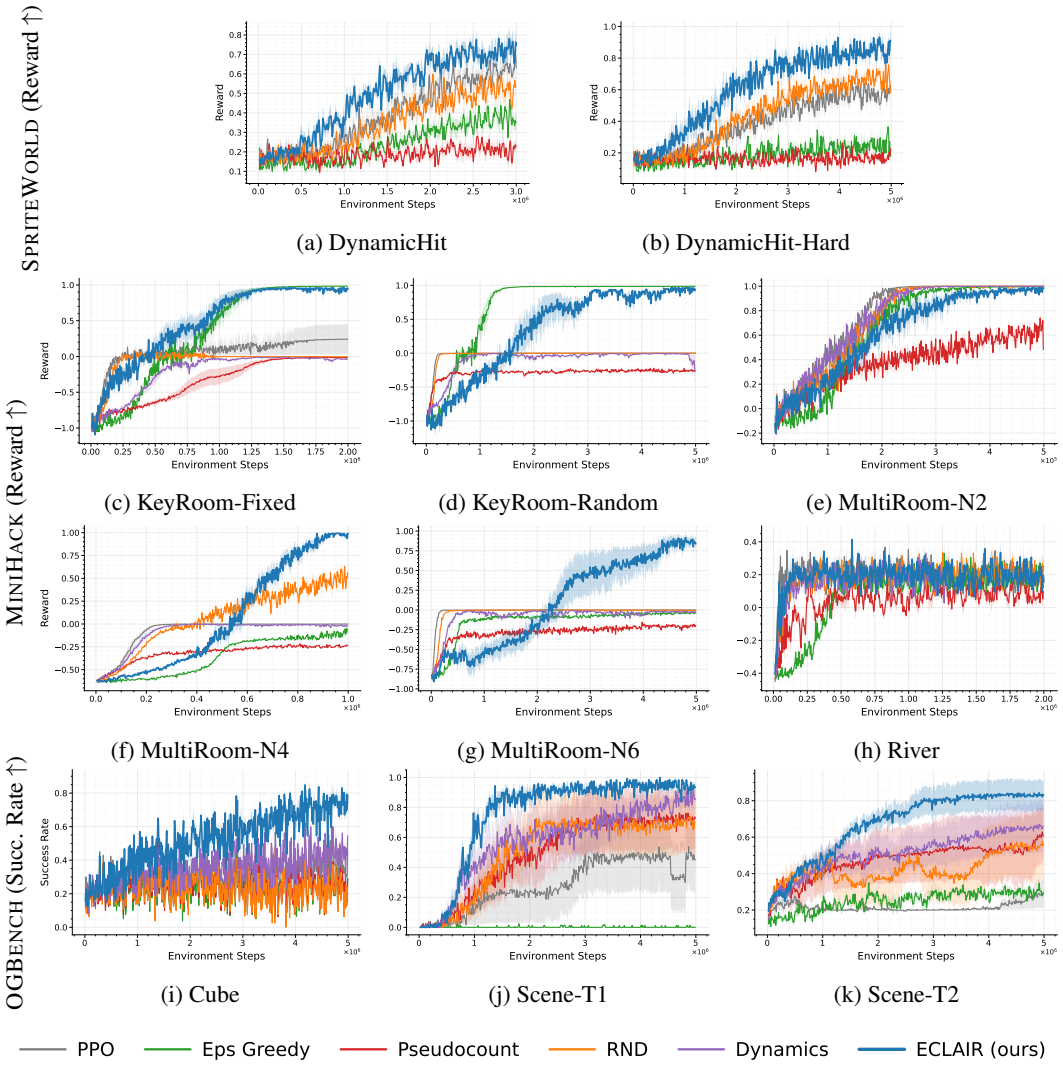


Figure 6: Training curves across SpriteWorld (normalized reward ↑), MiniHack (reward ↑), and OGBench (success rate ↑) environments. Solid lines show the mean performance over 4 seeds; shaded regions show ± one standard error. ECLAIR consistently outperforms all baselines across our diverse benchmark, with its most prominent gains on difficult long-horizon tasks.

`KeyRoom-Random` and `MultiRoom-N2`, ECLAIR reaches near-optimal performance slightly more slowly than the strongest baseline. We attribute this early lag to the additional representation-learning phase in ECLAIR: before the entity-centric dynamics bonus becomes fully informative, the DLP representation and latent prediction model must first adapt to the environment. This effect could likely be reduced with more task-specific tuning, particularly by sweeping the number of representation-model update steps.

On `River`, all methods quickly plateau at low reward, suggesting that the task requires additional capabilities beyond exploration bonuses, such as memory or more structured planning. In OGBench, ECLAIR obtains the best performance on `Cube` and converges faster and higher than the baselines on `Scene-T1` and `Scene-T2`. The gap is especially clear in the scene tasks, where the agent must discover multiple object interactions and execute them in the correct order. Across the full set of curves, the baselines exhibit strong domain dependence: ε-greedy performs well on several discrete MiniHack tasks but degrades in continuous-control manipulation, while prediction-based baselines

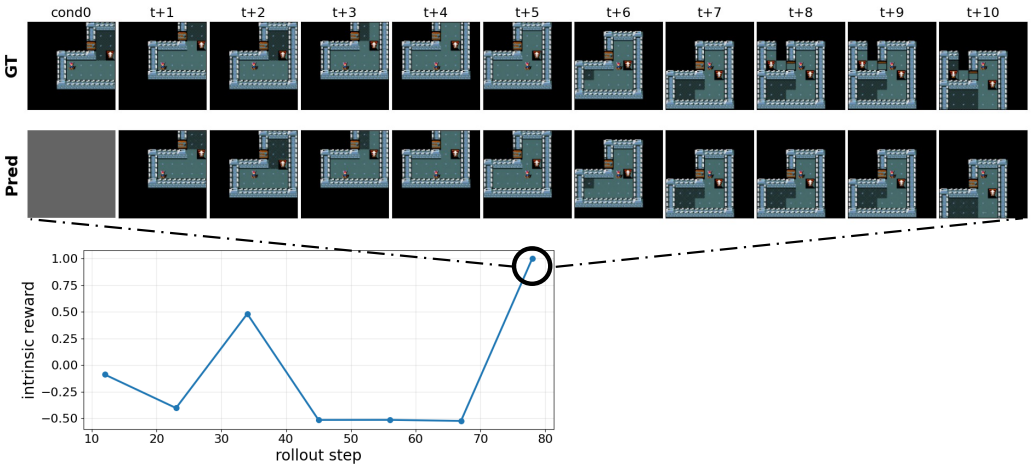


Figure 7: Trajectory segment from `KeyRoom` with a large ECLAIR intrinsic bonus. The top row shows the ground-truth rollout and the bottom row shows the corresponding dynamics predictions. The intrinsic reward peaks near the end of the segment, precisely when the agent completes a meaningful environment-changing interaction by opening the locked door. This suggests that ECLAIR assigns high exploration bonus to task-relevant transitions that are difficult to predict under the learned entity-centric dynamics model.

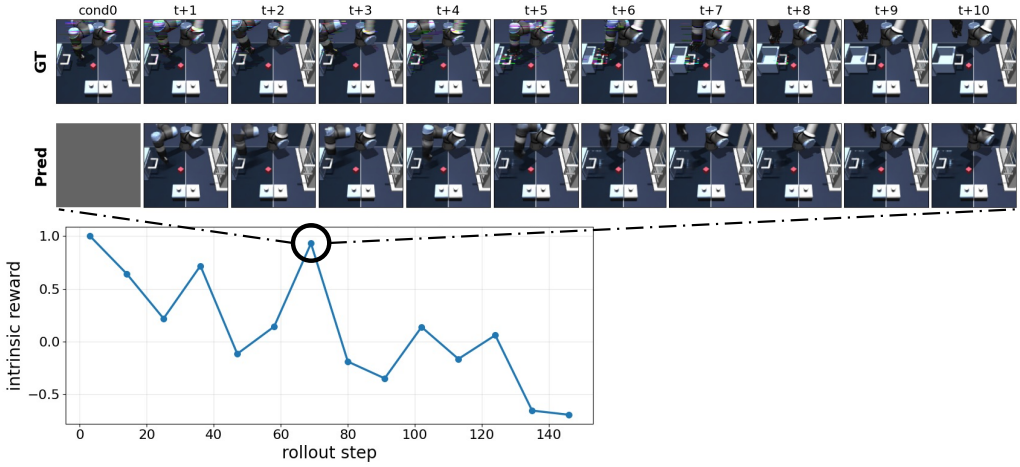
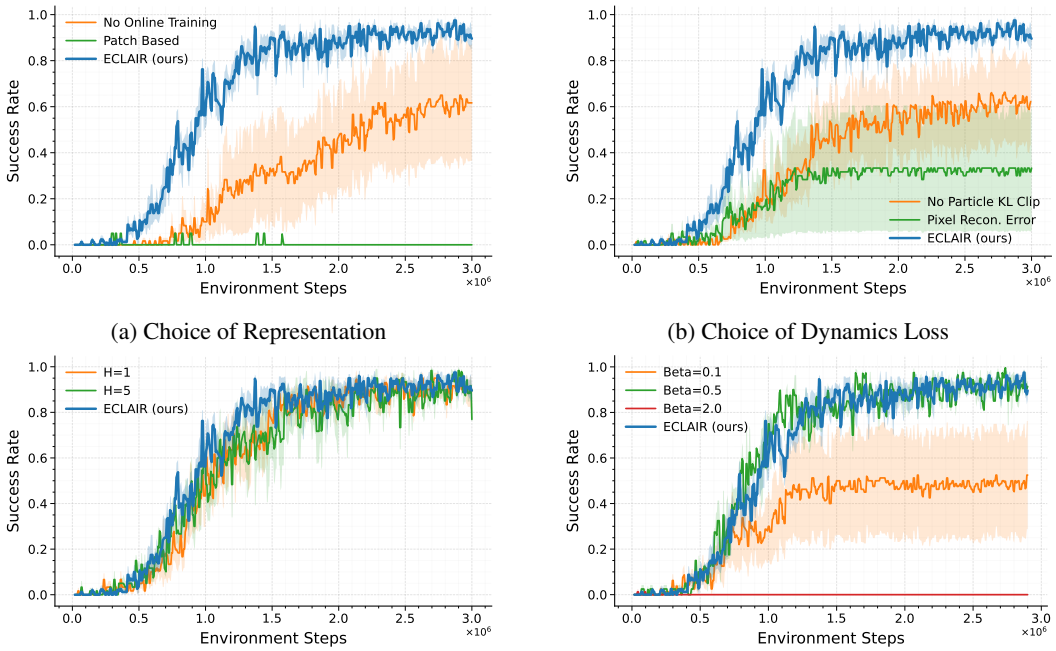


Figure 8: Trajectory segment from `Scene-T1` with a large ECLAIR intrinsic bonus. The top row shows the ground-truth rollout and the bottom row shows the corresponding dynamics predictions. The intrinsic reward spikes during the time of the trajectory when the agent opens the drawer. This suggests that ECLAIR assigns high exploration bonus to task-relevant transitions that are difficult to predict under the learned entity-centric dynamics model.

such as RND and Dynamics are more competitive in visually rich continuous domains but remain less consistent across tasks. These results support the main conclusion that ECLAIR provides a more reliable exploration signal across diverse sparse-reward visual environments.

We further show the training curves of ECLAIR against its ablations in Figure 9.

DLP decompositions. We visualize the DLP decompositions used across tasks in Figures 10–13. For each environment, we overlay the inferred particles on the original observation and show the corresponding foreground and background reconstructions from the particle representation.



(a) Choice of Representation

(b) Choice of Dynamics Loss

(c) Prediction Interval H

(d) Intrinsic reward scale β

Figure 9: Ablation studies on `Scene-T1`. We evaluate key design choices in ECLAIR, including the DLP representation, dynamics loss, prediction interval H, and intrinsic reward scale β.

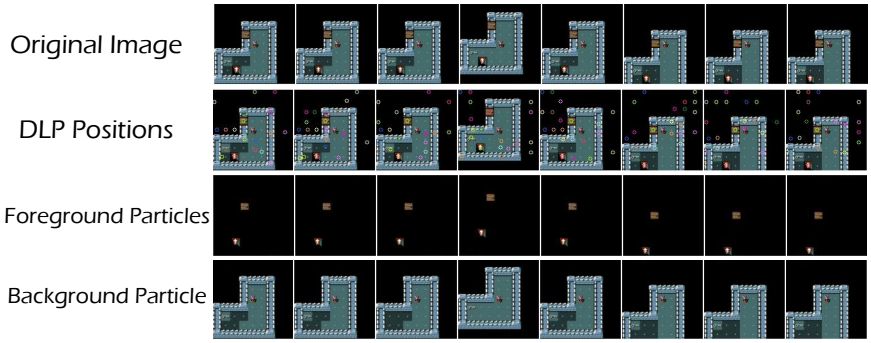


Figure 10: DLP decomposition examples on the MiniHack `KeyRoom` task. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

Alternative representation visualizations. To better understand the role of entity-centric structure in ECLAIR, we compare DLP with a non-object-centric patch representation in Figure 14. This patch-based variant uses a dynamics VAE (DVAE) adapted from LPWM, where the "particles" are replaced by fixed-grid patch embeddings. The number of patches is matched to the number of DLP particles, and the downstream Transformer dynamics model is kept the same, so the comparison isolates the effect of the representation rather than model capacity.

Unlike DLP, the patch-based representation does not explicitly model object attributes or maintain object-level slots. Instead, a CNN encoder down-samples each input image into a spatial grid of patch features. However, as shown in Figure 14, the learned patches primarily encode fixed spatial regions and fail to preserve fine-grained details of task-relevant objects, such as the robot arm and

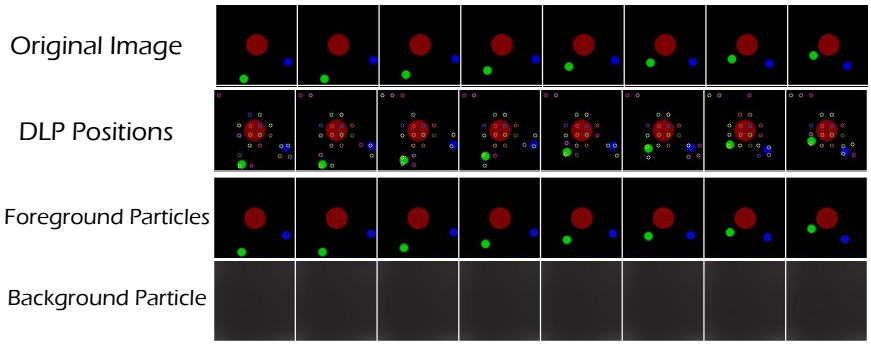


Figure 11: DLP decomposition examples on the SpriteWorld `DynamicHit` task. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

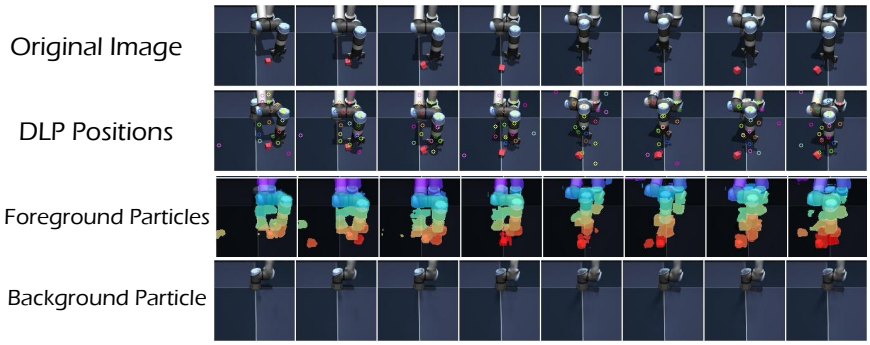


Figure 12: DLP decomposition examples on the OGBench `Cube` task. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

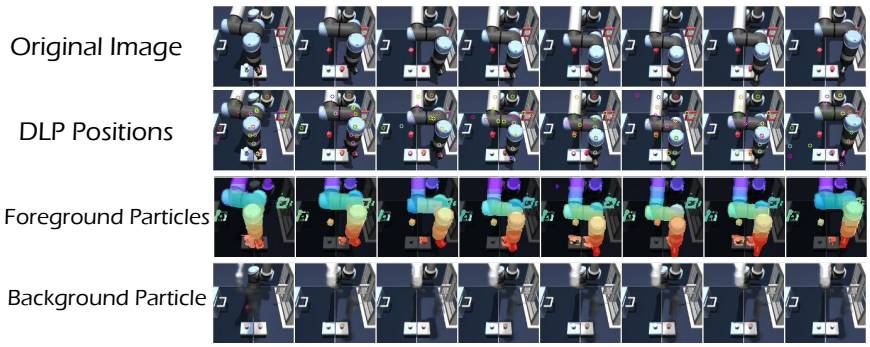


Figure 13: DLP decomposition examples on the OGBench scene manipulation tasks. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

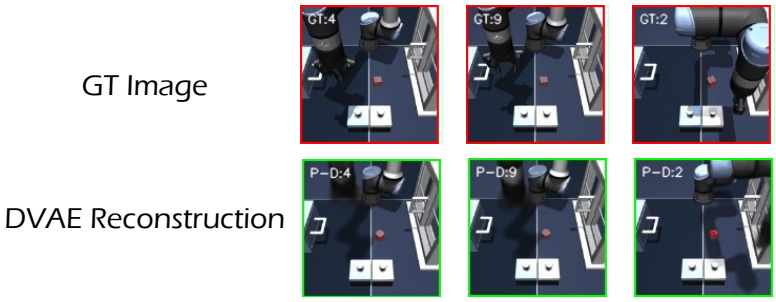


Figure 14: Visualization of the patch-based DVAE representation on the manipulation environment. Unlike DLP, the patch representation encodes fixed spatial regions rather than object-level entities, and therefore fails to preserve fine-grained details of task-relevant objects such as the robot arm and gripper.

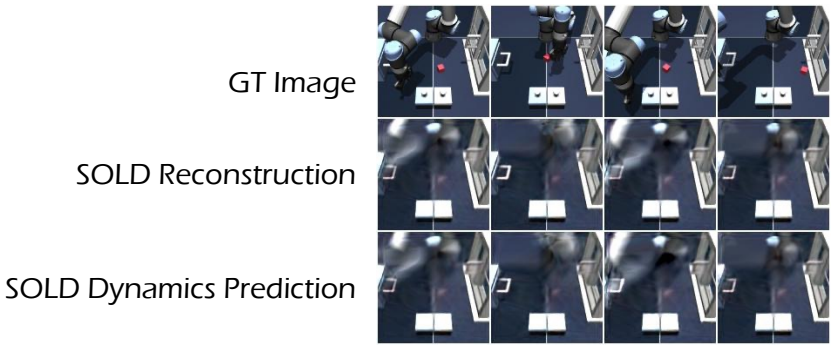


Figure 15: Visualization of Slot-based representation and SOLD dynamics prediction on the `Scene-T1` task. The model fails to learn meaningful reconstructions or dynamics prediction.

gripper. This weakens object-level prediction signal used for exploration and results in no meaningful learning progress in the `Scene-T1` task.

Additionally, we experimented with a Slot Attention-based representation (Locatello et al., 2020) and dynamics model, using SOLD (Mosbach et al., 2025). In this variant, we train the SOLD dynamics model for exploration. However, as shown in Figure 15, the slot-based model fails to learn meaningful reconstructions or dynamics prediction on the `Scene-T1` task. Further architecture adaptation and careful hyperparameter tuning may be required to obtain non-trivial slot-based representations in this setting, which we leave for future work.

G Broader Impacts

Our work aims to advance the ability of reinforcement learning agents to explore in visual environments with sparse rewards. Improved exploration can benefit the development of more capable autonomous agents, particularly in unstructured real-world settings where agents must discover useful interactions without dense supervision. Potential positive applications include robotics, assistive automation, scientific discovery, and other domains where agents must learn from limited feedback.

At the same time, improving autonomous exploration also raises safety concerns. Agents that explore more effectively may take unexpected actions, interact with objects in unintended ways, or discover strategies that are unsafe in real-world environments. These risks are especially important when deploying embodied agents in physical settings shared with humans. Therefore, methods for improved exploration should be paired with safeguards such as safe exploration constraints, human oversight, restricted action spaces, simulation-based validation, and monitoring mechanisms that detect unsafe or unintended behavior. Our experiments are conducted in simulated environments, but future deployment of similar methods should ensure that exploration is performed in a safe, controlled, and responsible manner.