ENTITY-CENTRIC EXPLORATION THROUGH LATENT DYNAMICS FOR VISUAL REINFORCEMENT LEARNING

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, 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. Through a theoretical connection between dynamics-prediction error and count-based exploration via information gain, we show that exploiting entity structure can reduce exploration complexity, which motivates our entity-centric approach to exploration. 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

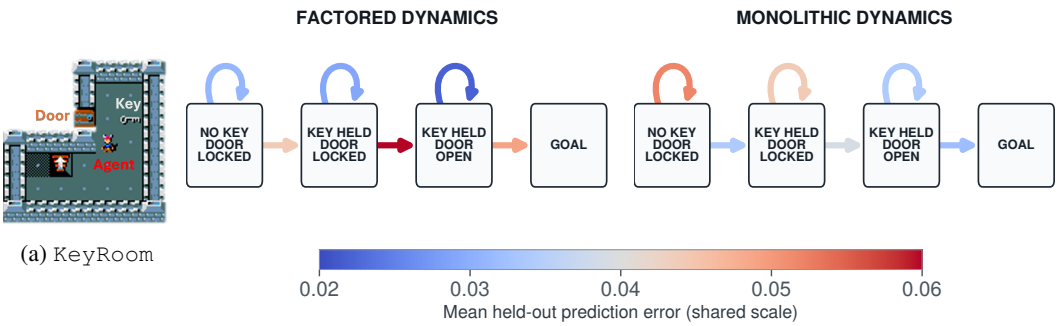
Reinforcement learning has made impressive strides across a wide range of domains, from games and logistics optimization to robotics and large language models (Silver et al., 2017; Rolf et al., 2023; Guo et al., 2025; Li et al., 2026). Exploration is essential to online reinforcement learning, yet not all experiences contribute equally to learning. Consider the sparse-reward `KeyRoom` task in Figure 1, where the agent must acquire a key, unlock a door, and then reach the goal to receive a reward. To show how entity structure affects the exploration signal, we train factored and monolithic dynamics models on the same interaction data and evaluate them on held-out transitions. The factored model assigns higher prediction error to task progress than to unchanged states, suggesting that entity structure can provide a good exploration signal. 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 a cluttered room, representing the scene as {`robot`, `drawer`, `shirt`, ...} 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 object-centric learning approaches (Daniel & Tamar, 2022; Locatello et al., 2020) that discover entities in an unsupervised manner.

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.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.



(a) KeyRoom

Figure 1: **Entity-centric prediction error highlights task progress.** *Left:* In KeyRoom, the agent must pick up the key, unlock the door, and reach the goal to receive a reward. *Right:* Abstract task states and their transitions, colored by mean dynamics prediction error. The factored dynamics model assigns higher error to task progress transitions than to the self-loop transitions, whereas the monolithic model has higher errors on self-loop transitions that do not leads to task progress.

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. We use it to motivate our method in the factored setting, 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 draw on the connection between dynamics prediction error and count-based exploration to motivate an entity-centric exploration signal for visual environments. Rather than estimating uncertainty over the full joint space, we consider novelty at the level of individual entities. For K entities, each with state space $\mathcal{E}$, this reduces the space over which novelty is estimated from $\Theta(|\mathcal{E}|^K)$ joint configurations to $\Theta(K|\mathcal{E}|)$ entity states. This structural reduction motivates exploiting entity structure for more efficient exploration in long-horizon tasks.

We present ECLAIR, which stands for **E**ntity-**C**entric **L**atent dyn**A**mics **I**ntrinsic **R**eward. ECLAIR combines an entity-centric representation with a dynamics model over entity latents and uses the sum of per-entity KL divergences between the encoder posterior and the dynamics prior as a structured exploration bonus. We jointly train the entity encoder, dynamics model, and RL policy online, allowing the representation to improve as the agent collects more informative experiences. We evaluate ECLAIR on 10 challenging visual RL tasks, where it achieves an average score of 0.90 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.

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:

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

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, $z_t = q_\phi(o_t) = \{e_t^1, \ldots, e_t^K\}$. Additional background on count-based exploration and object-centric representation learning is 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 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.

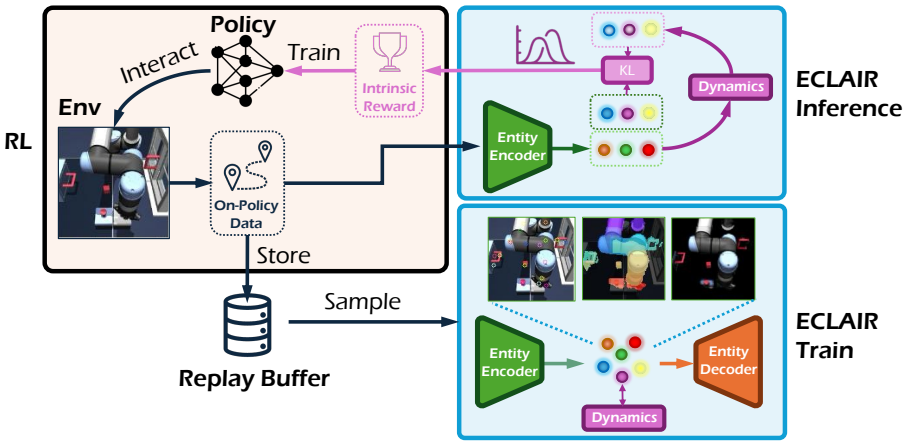


Figure 2: Overview of ECLAIR. ECLAIR learns an entity-centric representation and latent dynamics model online from the agent's experience. During inference, image observations are encoded into entity latents, and the KL divergence between the encoded posterior and the dynamics prediction is used as an 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.

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:*

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

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

Proposition 1 (Pseudo-count Proxy and Dynamics Prediction Error)**.** *Under Assumption 1, consider the fixed-variance mean-update approximation. For sufficiently small positive prediction gain, the local pseudo-count proxy and corresponding bonus satisfy*

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

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

In practice, we use $\mathrm{KL}(q_\phi(z' \mid s') \| p_\xi(z' \mid z, a))$ as the intrinsic bonus. Its mean term is proportional to squared prediction error normalized by the prior variance, and its variance term captures additional predictive mismatch. This KL also serves as a proxy for the information gain $\mathrm{IG}_n(s')$ about the next latent state. We provide full derivation in Appendix A.1.

Entity-centric structure can reduce exploration complexity. After connecting dynamics prediction error with pseudo-counts, we show that entity-centric reward structure can improve optimistic exploration in a bandit setting, as a simplified abstraction of exploration in an 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 and the sampling conditions in Appendix A.2, let $\Delta_k(e, a)$ denote the suboptimality gap of action a for entity k in state e. Suppressing logarithmic confidence factors, the UCB component sample complexity is bounded by*

$$N_{\text{fact}} = \sum_{k=1}^{K} \sum_{e \in \mathcal{E}_k} \sum_{a:\, \Delta_k(e,a)>0} \widetilde{O}\Big(\Delta_k(e,a)^{-2} \log\log\big(C_0\, \Delta_k(e,a)^{-2}\big)\Big), \tag{3}$$

where C_0 is a fixed numerical constant. There are KE component bandits versus E^K joint-state bandits, each with A arms. The explicit confidence dependence and proof are in Appendix A.2.

We also relax exact additivity by allowing approximate reward interactions among groups of entities. Appendix A.3 introduces this alternative reward structure and gives the corresponding tabular sample-complexity bound.

These results justify our using entity-centric prediction error as a practical proxy for pseudo-count exploration. While the analysis is simplified, the assumptions 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, ECLAIR jointly learns an entity-centric (EC) representation $q_\phi(o_t) = \{e_t^1, \ldots, e_t^K\}$ from a pixel observation o_t together with an EC 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, ECLAIR trains both components jointly so that the latent entities are directly shaped to be predictable by the dynamics model.

The EC 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:

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

The output distribution parameters serve as the prior in the KL divergence between the EC representation posterior and the EC 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:

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

Clipping prevents a single entity with unusually large prediction error from dominating the reward, since the KL is unbounded.

For our main experiments, we instantiate the EC representation using DLP (Daniel & Tamar, 2022; 2024) and the EC dynamics model using LPWM (Daniel et al., 2026). We evaluate variants using alternative EC representations and dynamics models in Section 5.3. 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.

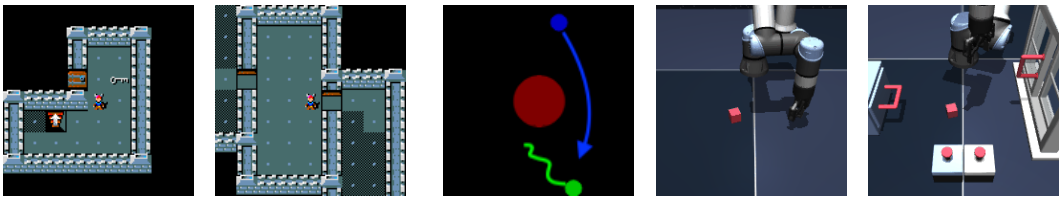
A second issue is that the intrinsic reward depends on the dynamics model's uncertainty, and this uncertainty is only informative when the model is updated on the same distribution induced by the current policy. If the dynamics model is trained only once and then fixed, its prediction error quickly becomes 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 (Schulman et al., 2017) as the base RL algorithm and maintain separate value heads for extrinsic reward V^e and intrinsic reward V^i, with separate Generalized Advantage Estimation (GAE) (Schulman et al., 2015) for each. The combined advantage is $\hat{A}_t = \hat{A}^e_t + \beta \hat{A}^i_t$, 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:

$$r^i_{t-H} = \frac{1}{H \cdot K} \sum_{h=0}^{H-1} \sum_{k=1}^{K} \mathrm{clip}\big(\ell^k_{t-H+h},\ 0,\ \mathrm{KL}_{\max}\big)$$

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) `DynamicHit` (d) `Cube` (e) `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)** Can ECLAIR leverage different variants of entity-centric representations and dynamics models? **(4)** 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

(a) MINIHACK (Reward ↑)

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

(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	0.75 ± 0.25	0.63 ± 0.18
RND	0.30 ± 0.04	0.70 ± 0.24	0.63 ± 0.18
Dynamics	0.48 ± 0.10	0.85 ± 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 best performing method(s).

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 10 tasks spanning three benchmarks, all providing minimal reward signal until the agent completes the final goal. MiniHack focuses on skill-based 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 five MiniHack tasks, `KeyRoom-Fixed`, `KeyRoom-Random`, and `MultiRoom-N2/N4/N6`; 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 we observed the original model struggled with our challenging benchmark. 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. All methods, including ECLAIR, use the same PPO backbone and the same pixel-based actor-critic, differing only in the exploration bonus. All methods are compared under the same environment interaction budget. We discuss computational cost and report additional compute-matched experiments in Appendix F.

Results. As shown in Figure 5, 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. 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.

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 en-

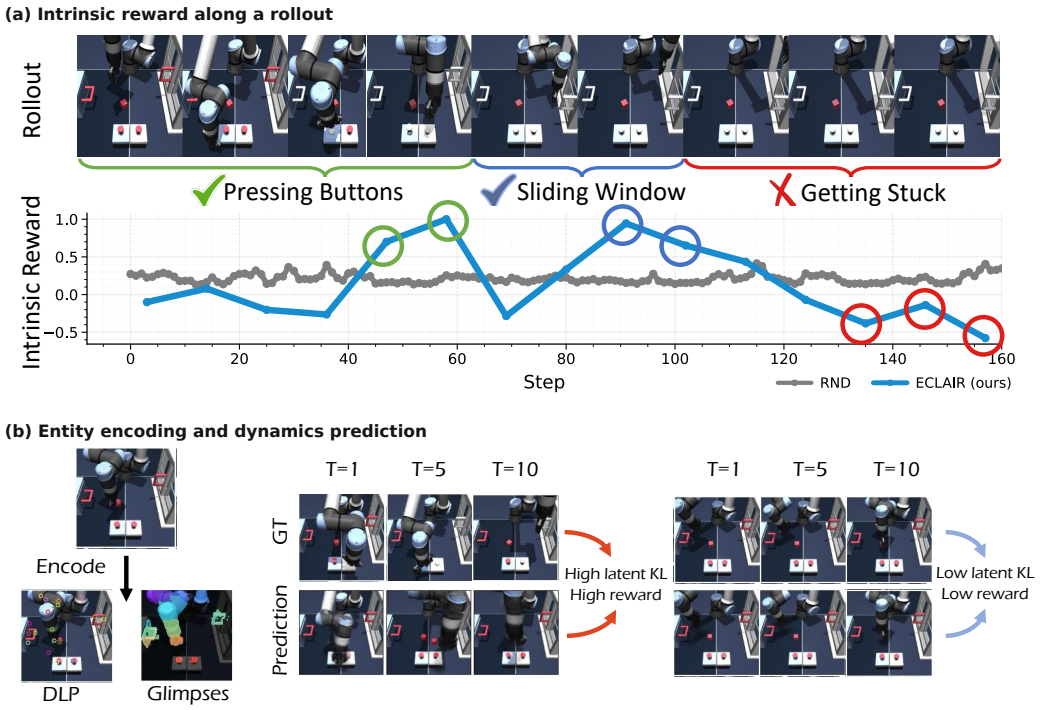


Figure 4: **Intrinsic reward and dynamics predictions on `Scene-T2`.** *(a)* Intrinsic rewards assigned by ECLAIR and RND along a rollout. *(b)* Entity encoding and decoded dynamics predictions for examples with high and low latent KL divergence. Ground-truth observations are shown above the predictions.

vironments. ECLAIR proves to be the method that outperforms the baselines across our diverse benchmarks. Additional results, including training curves, can be found in Figure 5 in Appendix F.

5.2 Analysis of the Intrinsic Reward

We visualize how ECLAIR produces its intrinsic reward during RL training. Figure 4(a) compares the intrinsic rewards assigned by ECLAIR and RND along a rollout on `Scene-T2`. ECLAIR assigns higher reward to task-progress interactions, such as pressing buttons and sliding the window, and lower reward when the agent becomes stuck. RND produces a more uniform signal along the same trajectory. To examine the source of this signal, Figure 4(b) visualizes ECLAIR's entity encoding and latent dynamics predictions. The high-reward example exhibits a larger mismatch between predicted and observed object states, while the low-reward example shows closer agreement. Together, these examples suggest that entity-centric dynamics prediction error provides a targeted exploration signal by highlighting meaningful object-state changes. Additional visualizations of ECLAIR's intrinsic rewards can be found in Appendix F.

5.3 Model Variants and Ablation Studies

Alternative representations and dynamics models. We investigate whether the ECLAIR exploration framework can support alternative representations by replacing the DLP encoder with a patch-based encoder on MultiRoom and computing the latent KL over the individual patch representations. We additionally evaluate a slot-attention (Locatello et al., 2020) based representation paired with the SOLD (Mosbach et al., 2025) dynamics model on the `Scene` tasks using latent dynamics prediction error. Table 2 compares these variants directly with ECLAIR with LPWM on the corresponding environments. The patch-based variant is identical to LPWM where each entity is encoded from patches rather than an object-centric decomposition; the slot-based variant replaces both the encoder and the dynamics model, and uses slot-space prediction error in place of the latent KL since SOLD's dynamics are deterministic.[1] Full details of both variants are in Appendix C.1.

On `MultiRoom-N4`, ECLAIR with LPWM achieves a higher reward than the patch-based variant, while both methods substantially outperforming the baseline using monolithic representation (Dynam-

[1]For deterministic latents, MSE is proportional to KL divergence between Gaussians with unit covariance.

Method	MultiRoom-N4	MultiRoom-N6	Scene-T1	Scene-T2
Dynamics (monolithic representation)	-0.02 ± 0.01	-0.02 ± 0.01	0.85 ± 0.10	0.63 ± 0.15
ECLAIR with Patch-Representation (ours)	0.89 ± 0.08	0.78 ± 0.04	–	–
ECLAIR with Slot-Representation (ours)	–	–	0.85 ± 0.03	$\mathbf{0.88 \pm 0.10}$
ECLAIR with LPWM (ours)	$\mathbf{0.99 \pm 0.00}$	$\mathbf{0.89 \pm 0.08}$	$\mathbf{0.90 \pm 0.04}$	$\mathbf{0.84 \pm 0.09}$

Table 2: Comparison of ECLAIR model variants. MultiRoom-N4 and MultiRoom-N6 report MiniHack reward and OGBench success rate. ECLAIR with LPWM is the full model and uses a DLP representation. The slot-based variant uses the SOLD (Mosbach et al., 2025) model. We report mean $\pm$ std. error over 4 seeds.

ics). These results show that patch representations can support effective exploration in MultiRoom, although they do not consistently match the full DLP-based model. On the Scene tasks, the slot-based variant remains competitive with LPWM. ECLAIR with LPWM peforms better on `Scene-T1` and comparably on `Scene-T2`. These results show that ECLAIR remains effective across multiple representation and dynamics choices, indicating that its entity-centric exploration through latent dynamics framework is not tied to a particular encoder or dynamics model.

Method ablations. We ablate the key design choices of ECLAIR with LPWM on `Scene-T1`, reporting success rate averaged over four seeds. All variants retain the DLP representation and LPWM dynamics architecture unless otherwise stated. and results are summarized in Table 3, and the corresponding training curves are provided in Figure 8 in the Appendix. Removing online finetuning lowers success from 0.90 to 0.61, showing that encoder adaptation is important. Replacing latent dynamics exploration bonus with Pixel reconstruction error reduces success to 0.33, while removing KL clipping yields 0.63, showing both the latent prediction objective and clipping contributes to performance. Finally, we vary the prediction horizon of ECLAIR and show that the performance remains stable across horizon 1, 5, and 10. We provide additional ablation results in Appendix F.

Method	Success Rate
Exploration	
w/o Online Finetune	0.61 ± 0.18
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 $H = 5$	$\mathbf{0.90 \pm 0.04}$
ECLAIR with LPWM (ours)	$\mathbf{0.90 \pm 0.04}$

Table 3: Ablation study on OGBENCH Scene-T1. Mean $\pm$ std error over 4 seeds (success rate $\uparrow$).

6 CONCLUSION

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

Limitations. Similarly to other dynamics-model-based exploration methods, ECLAIR introduces additional computational overhead due to the dynamics model training. In addition, ECLAIR treats the RL algorithm as a separate module and only modifies the exploration signal. Future work can explore tighter integration with policy optimization, such as off-policy RL or model-based planning. Finally, the quality of the learned entity-centric representation shapes ECLAIR's exploration signal. Our experiments cover partially observable scenes with occlusion and without clear object boundaries, and our alternative representations remain effective under imperfect decompositions. Future work can further quantify this relationship.

AI USE STATEMENT

In this work, we used generative AI tools for design or provide feedback on research methodology or experiments, implement methods, assist in the writing of proofs. We have not used generative AI tools for help develop theoretical models or conceptual frameworks, formulate mathematical claims, provide critical ingredients for proving mathematical claims, propose or refine hypotheses, support qualitative and thematic data analysis, interpret results, and generate synthetic data sets. Assist with translation, clean and reformat dataset, are not applicable to this work.

Additionally, we used generative AI tools to create or edit software code, suggest a structure for the paper, and correct typos and grammatical errors. We have reviewed all AI-assisted work. Multiple authors verified and tested LLM-generated code for correctness. We take responsibility for the final content of this work, including any text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This work involves no human subjects and releases no new datasets. All experiments are run in simulation on public benchmarks (MiniHack, Apache-2.0; SpriteWorld, Apache-2.0; OGBench, MIT). Because ECLAIR rewards unpredicted transitions, it deliberately pushes agents toward novel interactions; this is desirable in simulation, but on physical systems it should be paired with safe-exploration constraints, restricted action spaces, and human oversight. See Appendix G.

REPRODUCIBILITY STATEMENT

Appendix C gives the full training procedure (Algorithm 1), intrinsic-reward computation, and per-domain hyperparameters; Appendix C.1 covers the model variants, Appendix D the environments and evaluation protocol, and Appendix E the baselines. We will release open-sourced code upon publication.

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, 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.

Thomas Kipf, Gamaleldin F. Elsayed, Aravindh Mahendran, Austin Stone, Sara Sabour, Georg Heigold, Rico Jonschkowski, Alexey Dosovitskiy, and Klaus Greff. Conditional Object-Centric Learning from Video. In *International Conference on Learning Representations (ICLR)*, 2022.

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, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *arXiv preprint arXiv:1506.02438*, 2015.

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 construct a pseudo-count proxy motivated by the 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 density with $\rho'_n(z') := p_{\xi'_n}(z' \mid z, a)$, where ξ'_n are parameters after a single mean update of size α toward $q_\phi(z' \mid s')$, holding σ_ξ fixed, so that $\mu_{\xi'_n} = \mu_\xi + \alpha(\mu_\phi(s') - \mu_\xi(z, a))$.

Under Assumption 1, the KL divergence between the two Gaussians serves as a proxy for latent information gain (IG); it equals IG when the encoder is the exact Bayesian posterior corresponding to the dynamics prior. For equal variances ($\sigma_\phi^2 = \sigma_\xi^2$), the KL reduces to $\|\mu_\phi(s') - \mu_\xi(z, a)\|^2 / 2\sigma_\xi^2$. For the prediction gain, substituting the Gaussian log-densities and taking the expectation over $z' \sim q_\phi(z' \mid s')$ gives

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

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

Motivated by the small-gain pseudo-count approximation of Bellemare et al. (2016), define the proxy using the averaged prediction gain: $\hat{N}_n^{\mathrm{proxy}}(s') := (e^{\overline{\mathrm{PG}}_n(s')} - 1)^{-1}$. For small positive $\overline{\mathrm{PG}}_n$ and $\alpha \ll 1$, we obtain

$$\hat{N}_n^{\mathrm{proxy}}(s') \approx \overline{\mathrm{PG}}_n(s')^{-1} \approx \frac{\sigma_\xi^2}{\alpha \|\mu_\phi(s') - \mu_\xi(z, a)\|^2},$$

confirming that large normalized prediction error yields a small pseudo-count proxy, i.e. high novelty. The corresponding intrinsic bonus then satisfies

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

Finally, for equal variances, the KL is proportional to $(r^i_{\mathrm{proxy}})^2$ under the same small-gain approximation. With unequal variances, it additionally captures variance mismatch. This motivates using the full KL as a practical exploration bonus. □

A.2 Proof of Proposition 2

Component-bandit sampling model. Assume $A \geq 2$. A query to any (k, e, a) returns an independent observation in $[0, 1]$ with mean $r_k(e, a)$. Each of the KE component bandits has a unique best action $a^*(k, e)$, with gaps $\Delta_k(e, a) = r_k(e, a^*(k, e)) - r_k(e, a)$. The goal is to identify these component best actions, and N_{fact} counts component queries. This feedback and sampling access are assumptions in addition to additivity.

Run lil' UCB with fixed admissible parameters on each component bandit, calibrating its confidence parameter so the failure probability is at most $\delta' = \delta/(KE)$. By Jamieson et al. (2014, Theorem 2), each procedure returns its best arm and stops within

$$n_{k,e} = O\left(\sum_{a:\, \Delta_k(e,a) > 0} \frac{\log(2/\delta') + \log\log(\mathrm{e}/\Delta_k(e, a)^2)}{\Delta_k(e, a)^2}\right)$$

queries with probability at least $1 - \delta'$. A union bound makes all KE guarantees hold with probability at least $1 - KE\delta' = 1 - \delta$. Summing $n_{k,e}$ gives the upper bound

$$N_{\mathrm{fact}} = \sum_{k=1}^{K} \sum_{e \in \mathcal{E}_k} \sum_{a:\, \Delta_k(e,a) > 0} O\left(\frac{\log(2KE/\delta) + \log\log(\mathrm{e}/\Delta_k(e, a)^2)}{\Delta_k(e, a)^2}\right) \tag{5}$$

There are KEA component arms, compared with $E^K A$ tabular arms; actual sample costs also depend on the gaps. Suppressing the logarithmic confidence factor yields the main-text bound. The numerical constant C_0 can be fixed at 16, keeping the nested logarithm positive for every gap in $(0, 1]$. □

A.3 Relaxing additivity with approximate group rewards

We use the same tabular analysis as above, replacing individual entities by groups of at most m entities. The reward uses the same averaging convention as the original additive model.

Assumption 3 (Order-m Reward Structure)**.** *For a nonempty collection $\mathcal{C} \subseteq \{C \subseteq [K] : 1 \le |C| \le m\}$, let $G = |\mathcal{C}|$ and $\mathcal{E}_C = \prod_{k \in C} \mathcal{E}_k$. There exist group rewards $r_C : \mathcal{E}_C \times \mathcal{A} \to [0,1]$ such that*

$$\sup_{s,a} \left| \mathbb{E}[r(s,a)] - \frac{1}{G} \sum_{C \in \mathcal{C}} r_C(e^C, a) \right| \le \varepsilon_m. \tag{6}$$

Use the sampling conditions of Appendix A.2, with (k, e) replaced by (C, u): each group reward is separately observable and sampleable, with bounded observations and a unique best action for each group configuration. Write $\Delta_C(u,a) = \max_b r_C(u,b) - r_C(u,a)$.

Proposition 3 (Tabular UCB Complexity under Group Rewards)**.** *Under Assumption 3 and the sampling conditions above, there are*

$$M = \sum_{C \in \mathcal{C}} E^{|C|} \le \sum_{j=1}^{m} \binom{K}{j} E^j$$

A-armed group-configuration bandits. For $\delta \in (0,1)$, lil' UCB identifies all their best actions with probability at least $1-\delta$, using at most group-reward observations N_{group}:

$$N_{\text{group}} = \sum_{C \in \mathcal{C}} \sum_{u \in \mathcal{E}_C} \sum_{a:\, \Delta_C(u,a) > 0} O\left(\frac{\log(2M/\delta) + \log\log(\mathrm{e}/\Delta_C(u,a)^2)}{\Delta_C(u,a)^2} \right) \tag{7}$$

Proof. Run lil' UCB on each of the M tabular bandits with failure probability $\delta' = \delta/M$. The same per-bandit bound used in Appendix A.2 applies (Jamieson et al., 2014, Theorem 2). A union bound gives

$$\Pr(\text{any group-bandit guarantee fails}) \le \sum_{C \in \mathcal{C}} \sum_{u \in \mathcal{E}_C} \delta' = M\delta' = \delta.$$

Summing the per-bandit sample costs proves the result. Overlap between groups does not affect this union bound. □

At $m = 1$, taking the K singleton groups gives $G = K$, $M = KE$, and, when $\varepsilon_1 = 0$, the original additive setting. For fixed m, E, A, the number of group arms MA is polynomial in K; the tabular joint-state model has $E^K A$ arms. Sample costs additionally depend on the gaps. The guarantee concerns group-wise best-action identification; ε_m measures how accurately their averaged rewards represent the joint mean.

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 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.

ECLAIR is defined by the exploration objective of Section 4.2, not by a particular world model. It requires only an entity-centric representation of the observation together with an action-conditioned predictive model over those entity latents, and any model exposing that interface can supply the bonus. This section describes the instantiation used in our main experiments, which pairs DLP (Daniel & Tamar, 2022; 2024) with LPWM (Daniel et al., 2026). We chose it not because the framework depends on it, but because it produced good results most consistently across all three domains; Appendix C.1 details two alternative instantiations, which also yield usable exploration signals but are less consistent across environments.

Instantiation with DLP and LPWM. In this instantiation the entity latents are the full LPWM particle set,

$$z^t = \{z_t^1, \ldots, z_t^M, z_{\text{bg}}^t\},$$

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

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

Algorithm 1 ECLAIR: Entity-Centric Latent dynAmics Intrinsic Reward

Require: Policy π_θ, value heads V^e_ψ, V^i_ψ, entity-centric encoder q_ϕ, dynamics prior p_ξ
Require: Intrinsic reward interval H, PPO epochs E_{ppo}, world-model 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 ▷ World-model update
end for

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. Under this instantiation, the per-entity KL of Equation 4 takes the attribute-wise masked form used by LPWM, since the DLP posterior factorizes over particle attributes. In the standard VAE, each posterior attribute distribution is compared against a fixed prior; here 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

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

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 4: Domain-specific hyperparameters for ECLAIR.

Here, $\text{att} \in \{o, s, d\}$ 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

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

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

$$r^i_t = \frac{1}{M+1}\left(\sum_{m=1}^{M} \text{clip}(\ell^m_t, 0, \text{KL}_{\max}) + \text{clip}\Big(\ell^{\text{bg}}_t, 0, \text{KL}_{\max}\Big)\right) .$$

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. The following procedure applies to any world model used with ECLAIR; we describe it in terms of the DLP + LPWM instantiation, and Appendix C.1 notes where the alternatives differ. We first pre-train the world model on randomly collected interaction data to warm up both the entity 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 the model. 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 the world model requires a relatively high update-to-data ratio during online training. As the policy discovers new states and interactions, the 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 take 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 world-model 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 parameters directly when computing the dynamics KL and the corresponding intrinsic reward.

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 the world model 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 4. We tune rollout length, episode horizon, world-model warmup, and world-model update-to-data ratio by domain, since the representation and dynamics model must adapt at different rates across different domains.

C.1 ECLAIR Variants: Alternative Representations and Dynamics Models

As noted in Appendix C, ECLAIR requires only that the world model expose (i) a per-entity latent decomposition of the observation and (ii) an action-conditioned predictive distribution or point estimate over the next-step latents. Any model with that interface can supply the bonus. Beyond the DLP + LPWM instantiation used in our main experiments, we evaluate two further instantiations, reported in Table 2: a *patch-based* representation, which retains the same dynamics module but removes entity-centric structure from the encoder, and a *slot-based* representation, which replaces both the encoder and the dynamics model. Instantiation-specific hyperparameters are listed in Table 5.

	Patch-based (DVAE)	Slot-based (SOLD)
Environments	MiniHack `MultiRoom-N4/N6`	OGBench `Scene-T1/T2`
Encoder	CNN patch grid (DVAE)	SAVi (Kipf et al., 2022)
Dynamics	PINT (same as LPWM)	OCVP transformer (Mosbach et al., 2025)
Bonus	Patch KL	Open-loop slot-prediction MSE
Entities / slots	64 patches	10 slots
Latent dim per entity	8	128
Dynamics layers / heads	6 / 4	4 / 8
Dynamics width	256	256 token, 512 hidden
Context frames	3	3
Prediction steps	1	10
Warmup collection	50K steps	50K frames
Online update steps	20 / iter	30 / iter
Online learning rate	$1e{-4}$	$1e{-4}$
Online batch size	8	4
Intrinsic scale β	0.1	0.1
Bonus clipping	$\mathrm{KL}_{\max} = 20$ per patch	$[0, 5]$ after normalization

Table 5: Hyperparameters for the two alternative-representation variants of ECLAIR. Both variants keep the PPO backbone, the dual value heads, and the online bonus-normalization pipeline unchanged; they differ only in the world model and the form of the prediction error.

Patch-based representation (DVAE). This variant replaces the DLP encoder with the patch-based DVAE encoder from LPWM while holding the dynamics module fixed, isolating the contribution of entity-centric structure from that of the dynamics architecture. A convolutional encoder downsamples the 128×128 observation by a factor of 16, producing a fixed 8×8 grid of 64 patch latents of dimension 8. This matches the number of foreground particles and the latent dimension used by LPWM on MiniHack, and the downstream PINT dynamics module is identical, so capacity and dynamics architecture are held constant and the only change is whether latents are bound to entities or to fixed image regions.

Two properties of the DVAE latent shape the resulting bonus. First, it contains only visual features, with no position, scale, depth, or transparency attributes and no separate background particle; since there is no transparency mask, all 64 patches contribute equally to the KL, whereas LPWM masks inactive particles out of the explicit-attribute and feature terms (Appendix C). Second, the patch variant measures one-step surprise rather than the H-step open-loop surprise used with LPWM; we observe that a prediction horizon of 1 performs best for this representation.

Slot-based representation (SOLD). This variant replaces both the representation and the dynamics model. Observations are encoded with SAVi (Kipf et al., 2022) into 10 slots of dimension 128, and the dynamics are given by SOLD's action-conditioned OCVP transformer (Mosbach et al., 2025). Because this dynamics module is deterministic and emits point predictions rather than distributions over slots, no KL is available, so we use the slot-space prediction error directly,

$$r_t^i = \frac{1}{H \cdot K \cdot D} \sum_{h=1}^{H} \left\| \hat{s}_{t+h} - s_{t+h} \right\|^2,$$

where $\hat{s}$ are slots predicted open-loop from 3 context frames over a horizon of $H = 10$ and s are the slots obtained by encoding the observed frames. This is the analogue of ECLAIR's latent KL under a deterministic dynamics model, and the same quantity as the *Dynamics* baseline's bonus, computed in slot space rather than in a monolithic latent space.

Warmup follows SOLD's staged protocol rather than LPWM's joint objective, since SAVi and the dynamics predictor are trained on different targets: we pretrain SAVi on reconstruction alone, then train the dynamics predictor autoregressively on *detached* slots while SAVi continues to finetune. During RL both modules are finetuned online, at an update-to-data ratio matching the one LPWM uses on OGBench so that the two arms receive the same model-training budget.

The intrinsic bonus is normalized by the running standard deviation of the intrinsic reward then clipped to $[0, 5]$.

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. We report per-method computational cost and a compute-matched comparison in Appendix F.

D Environment Details

We consider a total of 10 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.
`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 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) [2]. 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 4.

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. 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` [3] . 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.

[2] `https://github.com/vwxyzjn/cleanrl`. CleanRL is released under the MIT License.
[3] `https://github.com/NM512/dreamerv3-torch`

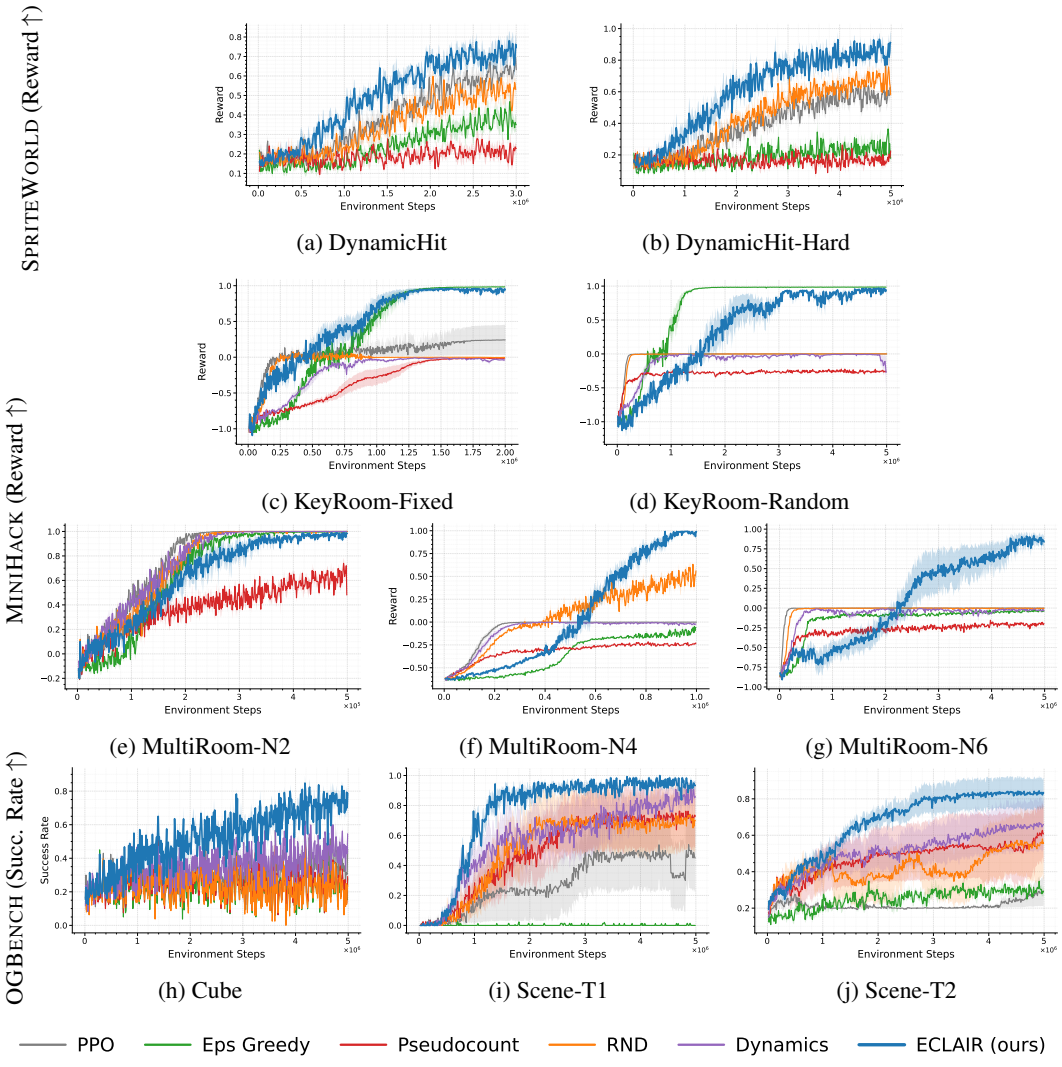


Figure 5: Training curves across SpriteWorld (normalized reward ↑), MiniHack (reward ↑), and OGBench (success rate ↑) environments. Solid lines show the mean performance over 4 seeds; shaded regions show ± one standard error. ECLAIR consistently outperforms all baselines across our diverse benchmark, with its most prominent gains on difficult long-horizon tasks.

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 6 and 7. Figure 6 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 7, 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.

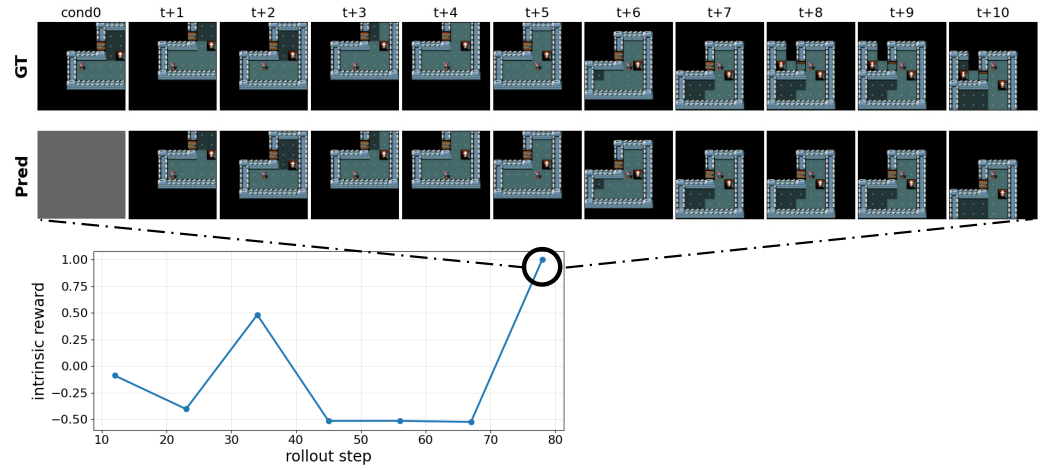


Figure 6: Trajectory segment from `KeyRoom` with a large ECLAIR intrinsic bonus. The top row shows the ground-truth rollout and the bottom row shows the corresponding dynamics predictions. The intrinsic reward peaks near the end of the segment, precisely when the agent completes a meaningful environment-changing 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.

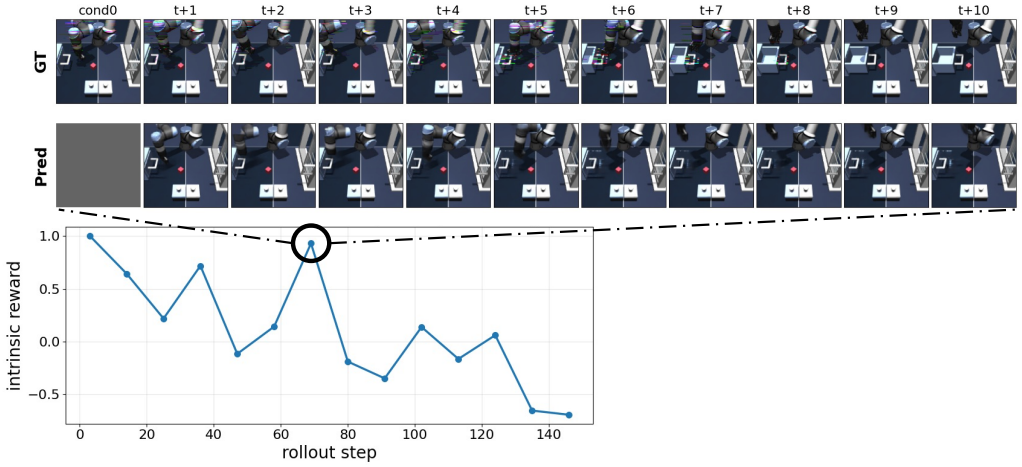


Figure 7: Trajectory segment from `Scene-T1` with a large ECLAIR intrinsic bonus. The top row shows the ground-truth rollout and the bottom row shows the corresponding dynamics predictions. The intrinsic reward spikes during the time of the trajectory when the agent opens the drawer. This suggests that ECLAIR assigns high exploration bonus to task-relevant transitions that are difficult to predict under the learned entity-centric dynamics model.

Training curves. Figure 5 shows the full training curves across all 10 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 `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.

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 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 8.

	Cost (5M steps)		Reward ↑	
Method	Wall-clock	GPU-hours	@ 5M env steps	@ $\sim$50 GPU-hrs
RND	10h	10	-0.01 ± 0.01	0.50 ± 0.29
Dynamics (DreamerV3)	17h	68	-0.02 ± 0.01	-0.02 ± 0.01
ECLAIR (ours)	12h	48	$\mathbf{0.89 \pm 0.08}$	$\mathbf{0.89 \pm 0.08}$

Table 6: Computational cost and compute-matched comparison on `MultiRoom-N6`. Wall-clock time and GPU-hours are measured for the standard 5M-environment-step runs; GPU-hours are wall-clock time multiplied by the number of GPUs used. The last column re-evaluates each method under a matched budget of approximately 50 GPU-hours rather than a matched number of environment steps. Mean $\pm$ standard error over 4 seeds.

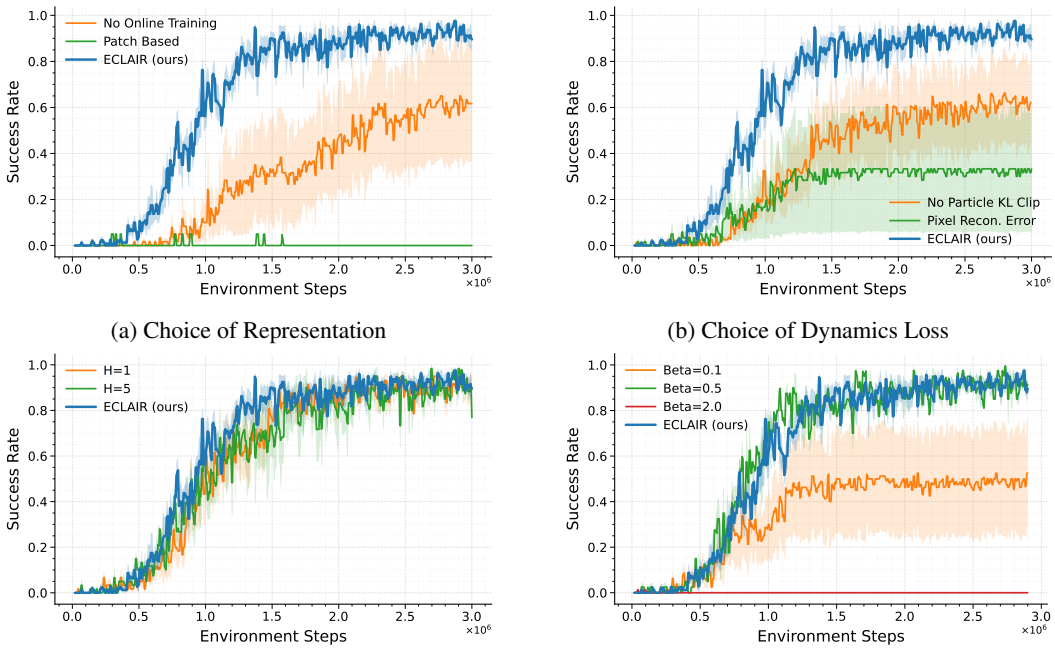
Computational cost and compute-matched comparison. We fix the environment interaction budget across all methods because interaction is the binding constraint in online RL: environment steps are inherently serial and, in any real deployment, the dominant cost, whereas model training parallelizes across devices and scales with available hardware. Sample efficiency is therefore the quantity of interest, and GPU-hours the cheaper resource to spend.

ECLAIR does spend more of that cheaper resource, since it trains an entity-centric dynamics model online. Dynamics (DreamerV3) receives the same model-training budget and in fact uses more GPU-hours than ECLAIR, so that comparison is already compute-matched. To also control for RND's much lower cost, we extend it to 25M environment steps on `MultiRoom-N6`, matching ECLAIR's $\sim$50 GPU-hours (Table 6). Even with $5\times$ the compute and $5\times$ the interaction, RND reaches only 0.50 ± 0.29 versus ECLAIR's 0.89 ± 0.08, with far higher seed variance. The gap therefore reflects the exploration signal rather than compute. ECLAIR's overhead is also not intrinsic: it is dominated by the transformer dynamics model, which we did not optimize for throughput.

DLP decompositions. We visualize the DLP decompositions used across tasks in Figures 9–12. For each environment, we overlay the inferred particles on the original observation and show the corresponding foreground and background reconstructions from the particle representation.

Alternative representation visualizations. We visualize the two alternative representations of Appendix C.1 to relate what each one encodes to how well it supports exploration.

Figure 13 shows the patch-based DVAE representation on MiniHack `MultiRoom`. Unlike DLP, it does not bind latents to objects: each latent describes a fixed image region regardless of what occupies it. On this task that is largely sufficient, since the grid-world decomposes cleanly into patch-level entities, and the model produces good reconstructions and reasonable dynamics predictions. Accordingly, this variant supports effective exploration, reaching 0.89 on `MultiRoom-N4` and 0.78 on `MultiRoom-N6` (Table 2), far above the monolithic Dynamics baseline: where the relevant events are localized appearance changes, patch-level surprise is a reasonable proxy for entity-level surprise. It nonetheless trails LPWM on both environments.



(a) Choice of Representation

(b) Choice of Dynamics Loss

(c) Prediction Interval H

(d) Intrinsic reward scale β

Figure 8: 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.

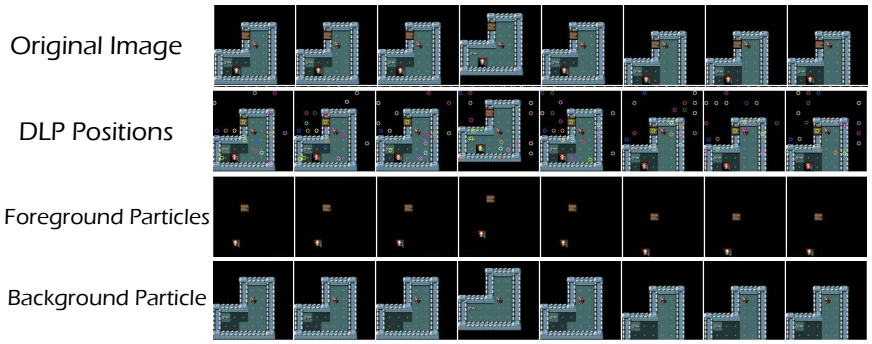


Figure 9: DLP decomposition examples on the MiniHack `KeyRoom` task. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

Figure 14 shows the slot-based SAVi (Kipf et al., 2022) representation and the corresponding SOLD (Mosbach et al., 2025) dynamics predictions on `Scene-T1`, using 10 slots. The decomposition preserves most of the important entities, and open-loop slot predictions remain informative enough to drive exploration, reaching 0.85 on `Scene-T1` and 0.88 on `Scene-T2`. Its reconstructions and predictions are nonetheless of lower visual quality than LPWM's, which does not appear to greatly affect the exploration signal: what matters for the bonus is that the important entities are represented and their transitions predicted, not that the decoded images are sharp.

All three instantiations therefore yield a usable exploration signal, and all substantially outperform a monolithic latent, confirming that the framework does not depend on a particular world model. What distinguishes them is consistency rather than peak performance: each alternative is competitive on

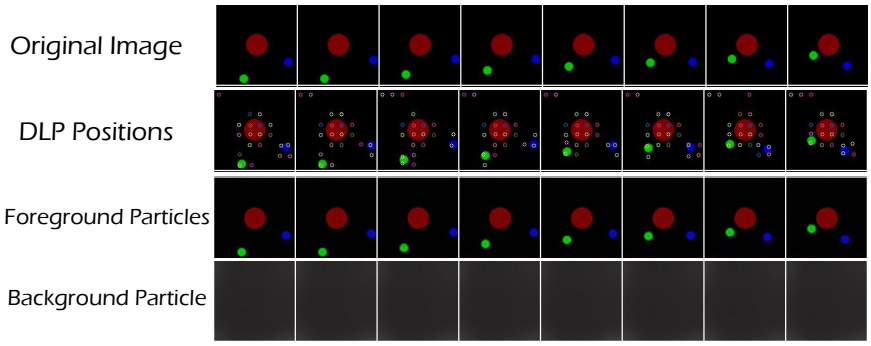


Figure 10: DLP decomposition examples on the SpriteWorld `DynamicHit` task. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

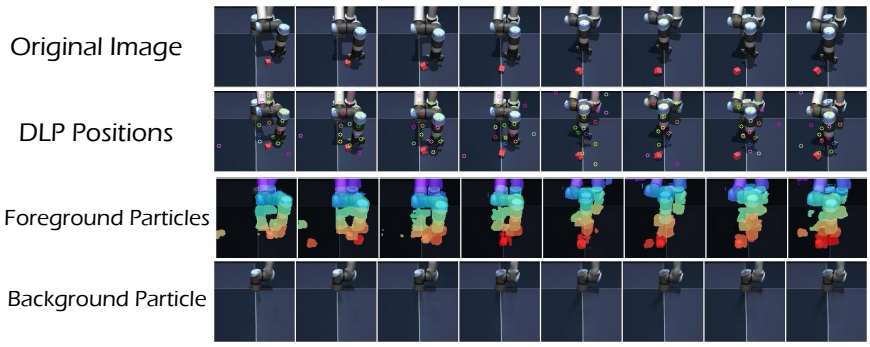


Figure 11: DLP decomposition examples on the OGBench `Cube` task. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

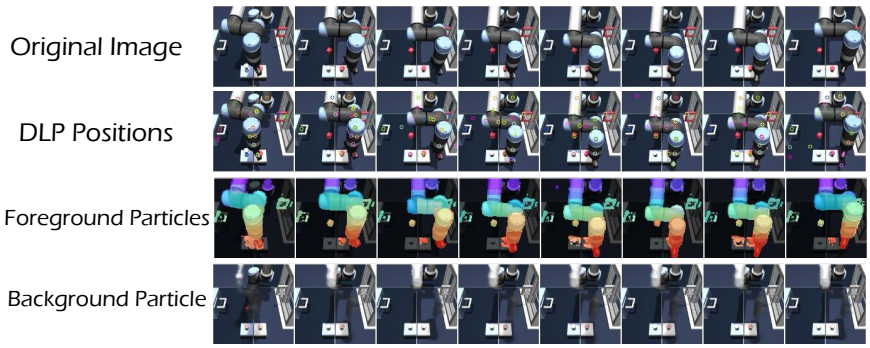


Figure 12: DLP decomposition examples on the OGBench scene manipulation tasks. Each example shows the original observation with inferred particles overlaid, together with the reconstructed foreground and background components.

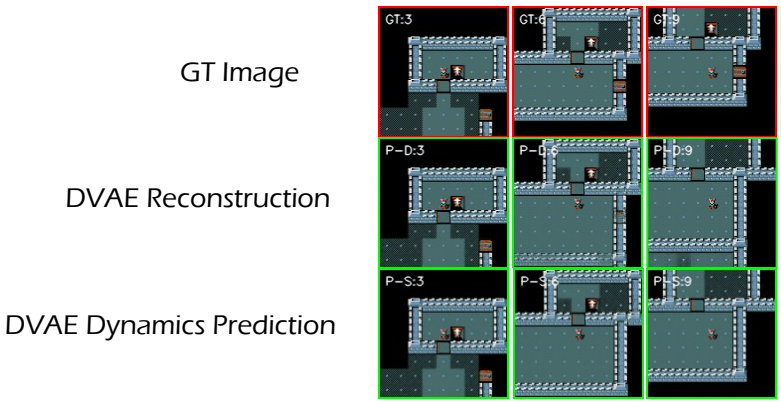


Figure 13: Patch-based DVAE representation and dynamics predictions on the MiniHack `MultiRoom` task. Unlike DLP, the patch representation encodes a fixed 8×8 grid of image regions rather than object-level entities. On this task that is sufficient: the grid-world decomposes cleanly into patch-level entities, and the model produces good reconstructions and reasonable dynamics predictions.

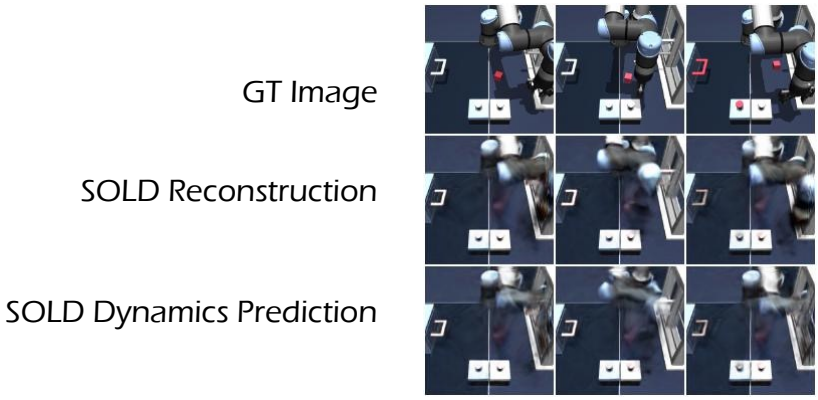


Figure 14: Slot-based representation and SOLD dynamics predictions on the `Scene-T1` task, using 10 slots. The decomposition preserves most of the important entities, though the reconstructions and predictions are of lower visual quality than LPWM's. This does not appear to greatly affect the resulting exploration signal (Table 2).

the domain it was run on, but DLP + LPWM matches or exceeds both on every environment where they are comparable, which is why we adopt it for the main experiments.

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.