# OBJECT-CENTRIC UNSUPERVISED SKILL DISCOVERY WITH ACTION INFLUENCE WEIGHTING

**Anonymous authors**
Paper under double-blind review

## ABSTRACT

Unsupervised skill discovery learns reusable behaviors without extrinsic task rewards. In multi-object environments, however, conventional skill representations typically encode behavior in a global state space, leaving unspecified which object a skill is intended to affect. This ambiguity can cause skill learning to favor easily induced changes in the agent or other state factors rather than meaningful changes in a target object. We address this problem with an object-centric skill representation $\zeta = (g, z)$, where the discrete variable $g$ selects a target object and $z$ specifies its continuous skill vector. Furthermore, we weight the object-level intrinsic reward by a causal action influence, which quantifies the conditional influence of actions on the target object's next-state distribution given the full current state. The resulting intrinsic objective focuses distance-maximizing skill learning on target-object state changes that are more controllable by the agent. Experiments across multiple object-interaction environments show that the proposed method improves object-state-space coverage, target-object adherence, and the utility of learned skills for downstream control compared with existing unsupervised skill discovery methods.

## 1 INTRODUCTION

Reinforcement learning commonly relies on task-specific reward functions, which can be difficult to design for long-horizon tasks and environments involving interactions with multiple objects (Booth et al., 2023). Sparse rewards further hinder exploration, as agents may require extensive interactions before receiving informative feedback. Unsupervised skill discovery addresses these challenges by learning a diverse repertoire of reusable behaviors through pretraining without extrinsic rewards (Achiam et al., 2018; Laskin et al., 2021). These skills can subsequently serve as low-level behavioral primitives for exploration and hierarchical control (Nachum et al., 2018; Sutton et al., 1999). A classical paradigm conditions the policy on a latent skill variable, encouraging different skill values to induce distinct behaviors or state distributions. Early information-theoretic and discriminative approaches learn distinguishable skills from visited states or transitions (Gregor et al., 2016; Eysenbach et al., 2018; Warde-Farley et al., 2018; Sharma et al., 2019).

Distance-maximizing skill discovery further provides a geometric interpretation of continuous skills by associating each skill vector with a direction in a learned state representation and training the policy to induce representation changes aligned with that direction (Park et al., 2022). These approaches provide a useful mechanism for learning diverse and reusable behaviors without specifying downstream tasks. In multi-object environments, however, defining skill directions over a holistic state representation creates an ambiguity that is not addressed by the skill variable itself. Let the state consist of an agent and multiple object factors. A global displacement objective can be satisfied by changing any component of the state, including the agent or objects unrelated to the intended behavior. In practice, some state factors may be substantially easier to change than others (Park et al., 2023), allowing the policy to obtain large intrinsic rewards without producing meaningful changes in a particular object (Carr et al., 2025). The resulting skill therefore specifies how the global state should change, but not which object should account for that change. This ambiguity is especially problematic for downstream hierarchical control, where a high-level policy must select skills according to object-specific task requirements.



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.

This observation suggests two requirements for skill discovery in multi-object environments. First, a skill should explicitly identify its target object so that its behavioral scope is well defined. Second, the learning objective should account for whether the agent can influence that object from the current state. Recent factorized skill discovery methods introduce state-factor structure, but typically assign skill components to multiple factors or combine factor-wise objectives into compositional behaviors (Hu et al., 2024; Wang et al., 2024; Hosseini & Baghshah, 2026). In contrast, we seek a single-target skill interface in which one skill execution specifies one object and one desired direction of change. Moreover, transition-likelihood-based objectives characterize the difficulty or likelihood of state transitions, but do not directly measure how strongly different actions influence the next-state distribution of a particular object (Park et al., 2023; Hosseini & Baghshah, 2026).

We therefore propose an object-centric formulation of unsupervised skill discovery in which each skill is represented as $\zeta = (g, z)$. The discrete variable $g$ selects a target object, while the continuous variable $z$ specifies a direction in that object's learned representation space. The policy receives the full state as context, whereas the intrinsic objective is defined only on the selected object. To account for state-dependent action influence, we introduce object-level causal action influence (CAI) (Seitzer et al., 2021), which quantifies the dependence of the target object's next-state distribution on the agent's action, conditioned on the current full state. As illustrated in Figure 1, CAI distinguishes states with different degrees of action influence over the target object. Specifically, we use the estimated $\mathrm{CAI}_i$ to weight the object-level intrinsic reward, encouraging skills to induce representation changes in states where the agent has greater influence over the target object.

Our main contributions are summarized as follows:

- We propose an object-conditioned extension of distance-maximizing skill discovery that explicitly separates target-object selection from continuous skill behavior.
- We introduce CAI-based intrinsic reward weighting to encourage controllable, directed state changes in target objects whose next states are more strongly influenced by the agent's actions.
- Experiments across multiple object-interaction environments demonstrate improvements in object-level state-space coverage, target-object adherence, and task relevance, as well as superior downstream task performance compared with existing skill discovery methods.

## 2 Related Work

### 2.1 Unsupervised Skill Discovery

Unsupervised skill discovery (USD) learns a skill-conditioned policy $\pi(a \mid s, z)$ through pretraining without extrinsic task rewards, where the latent variable $z$ represents distinct and reusable behaviors. Two prominent approaches are mutual-information-based skill discovery, which encourages skill distinguishability, and distance-maximizing skill discovery, which promotes diverse state-space displacements.

#### 2.1.1 Mutual-information-based Skill Discovery

Mutual-information-based methods maximize the dependence between the skill variable $Z$ and the visited-state variable $S$, encouraging different skills to induce distinguishable state distributions. Their objective is formulated as $I(S; Z) = D_{\mathrm{KL}}\left(p(s, z) \,\|\, p(s)p(z)\right) = H(Z) - H(Z \mid S)$. Since the true posterior $p_\pi(z \mid s)$ is generally intractable, a variational discriminator $q_\psi(z \mid s)$ is introduced to derive the lower bound $I(S; Z) \geq \mathbb{E}_{(s,z)\sim p_\pi}\left[\log q_\psi(z \mid s) - \log p(z)\right]$. This variational principle underlies early information-theoretic approaches to intrinsic control and skill discovery (Mohamed & Jimenez Rezende, 2015; Gregor et al., 2016; Eysenbach et al., 2018). Subsequent work extends this principle beyond state-level distinguishability by incorporating transition dynamics, state coverage, successor representations, and contrastive behavior representations (Sharma et al., 2019; Campos et al., 2020; Hansen et al., 2019; Liu & Abbeel, 2021; Laskin et al., 2022; Yang et al., 2023). Later methods further improve state-space coverage through incremental or guided skill discovery (Kamienny et al., 2021; Kim et al., 2023). These extensions enrich the behaviors that can be distinguished by the skill variable, but they do not specify which component of a multi-object state should account for the learned behavior.

### 2.1.2 DISTANCE-MAXIMIZING SKILL DISCOVERY

Distance-maximizing skill discovery (DSD) incorporates state-space geometry into skill learning. This family of methods can be interpreted through the Wasserstein dependency measure, defined as $I_W(S;Z) = W(p(s,z),\, p(s)p(z))$, where $W(\cdot,\cdot)$ denotes the Wasserstein distance (Ozair et al., 2019). For continuous skills, DSD learns a state representation $\phi : \mathcal{S} \to \mathbb{R}^d$ and aligns its displacement with a skill direction $z \in \mathbb{S}^{d-1}$. A representative formulation is (Park et al., 2022)

$$\begin{aligned} \sup_{\pi,\phi} \quad & \mathbb{E}_{z\sim p(z),\, \tau\sim p_\pi(\cdot|z)} \left[ \sum_{t=0}^{H-1} \big(\phi(s_{t+1}) - \phi(s_t)\big)^\top z \right] \\ \text{s.t.} \quad & \|\phi(s) - \phi(s')\|_L \le d(s,s'), \qquad \forall (s,s') \in \mathcal{S}_{\text{adj}} \end{aligned} \tag{1}$$

where $\mathcal{S}_{\text{adj}}$ denotes the set of adjacent state pairs, $\|\cdot\|_L$ is a prescribed norm, and $d(s,s') \ge 0$ bounds the representation distance. The objective promotes skill-dependent representation changes while constraining the geometry of the learned representation.

Park et al. (2022) introduce Lipschitz-constrained skill discovery to learn continuous directional skills under Euclidean distance constraints. Park et al. (2023) extend this framework with a learned, policy-dependent distance derived from transition likelihoods, emphasizing transitions that are difficult to achieve with the current skill repertoire. Park et al. (2024) further introduce metric-aware abstraction, using temporal distances to improve scalability to high-dimensional environments.

Despite these advances, these methods primarily define skill directions in a holistic state representation, without an explicit mechanism for specifying which object a skill should affect.

## 2.2 OBJECT-CENTRIC LEARNING IN REINFORCEMENT LEARNING

Object-centric learning represents environments as entities and their relations rather than holistic feature vectors. Interaction Networks and structured world models explicitly model object-level dynamics (Battaglia et al., 2016; Kipf et al., 2019), while locally factored and causal dynamics capture structured dependencies among state variables and actions (Pitis et al., 2020; Wang et al., 2022). Slot Attention further learns object-centric representations directly from visual inputs (Locatello et al., 2020). Such representations have been applied to exploration and manipulation through object-centric dynamics, planning, and exploration objectives (Watters et al., 2019; Sancaktar et al., 2022; Wang et al., 2023; Ferraro et al., 2025).

Recent work further introduces factor and interaction structure into skill discovery. Structured and counterfactual objectives identify interactions between factors (Choi et al., 2024; Chuck et al., 2025), while DUSDi aligns skill components with individual factors (Hu et al., 2024). SkiLD incorporates interactions among factors (Wang et al., 2024), and SUSD extends factorized skills to continuous directions with transition-likelihood weighting (Hosseini & Baghshah, 2026). Automatic task generation provides another route to interaction-oriented manipulation skills (Jansonnie et al., 2024). While these methods exploit factorized or interaction-aware structure, they rely on compositional skills, interaction objectives, or generated tasks. In contrast, our method separates target-object selection from skill behavior, with each execution specifying a single target object and a continuous object-level skill.

# 3 PROBLEM FORMULATION

We consider a Markov Decision Process (MDP) (Kearns & Koller, 1999) $\mathcal{M} = (\mathcal{S}, \mathcal{A}, r, p)$, where $\mathcal{S}$ and $\mathcal{A}$ denote the state and action spaces, $p(s' \mid s, a)$ is the transition probability, and $r : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is the extrinsic reward function. We focus on reward-free pretraining, where the skill-conditioned policy is learned without access to $r$. We consider a factored state space $\mathcal{S} = \mathcal{S}_1 \times \cdots \times \mathcal{S}_N$, where each state $s = (s_1, \ldots, s_N)$ consists of $N$ factors, with $s_i \in \mathcal{S}_i$ representing the state of the $i$-th factor. Each skill is defined as $\zeta = (g, z)$, where $g \in \{e_1, \ldots, e_N\} \subset \{0,1\}^N$ is a one-hot object-selection variable and $z \in \mathcal{Z} \subseteq \mathbb{R}^D$ is a continuous skill variable. Specifically, $g = e_i$ selects object $i$ as the target, while $z$ specifies the desired direction of change in that object's representation space. The skill-conditioned policy is denoted by $\pi_\omega(a \mid s, g, z)$. At the beginning of each skill segment of length $H$, $g \sim p(g)$ and $z \sim p(z)$ are sampled independently and remain fixed throughout the

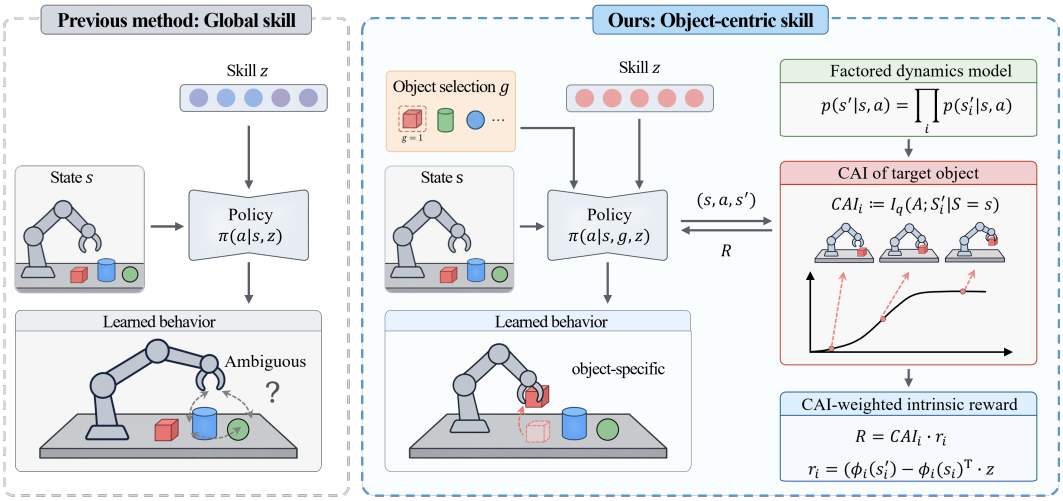



Figure 1: Overview of OCUSD. Unlike global skill discovery, OCUSD represents each skill as $(g, z)$, where $g$ selects a target object and $z$ specifies a continuous skill direction. A learned factorized dynamics model is used to estimate each object's causal action influence, which weights the object-level intrinsic reward and encourages meaningful object-specific skills.

segment. The resulting trajectory is denoted by $\tau = (s_0, a_0, s_1, \ldots, a_{H-1}, s_H) \sim p_{\pi_\omega}(\tau \mid g, z)$. Our objective is to learn a policy $\pi_\omega$ that induces distinct, direction-dependent changes in the state representation of the object selected by $g$.

## 4 Method

Figure 1 illustrates our proposed framework. We extend DSD to an object-conditioned formulation, where each skill $\zeta = (g, z)$ selects a target object through $g = e_i$ and specifies its skill behavior through $z$. Unlike factorized skill formulations that assign a separate continuous variable $z_i$ to every state factor, our method samples a single $z$ for the selected object for each skill execution. We optimize the objective under a unit adjacent-edge constraint on the target object's representation and weight the object-level intrinsic reward by its causal action influence $\mathrm{CAI}_i$. The object representations, factorized dynamics model, and skill-conditioned policy $\pi_\omega(a \mid s, g, z)$ are jointly optimized.

### 4.1 Object-Centric $(g, z)$-DSD

We decompose the state space of a multi-object environment into $N$ object-level factors. During pretraining, $g \sim p(g)$ and $z \sim p(z)$ are sampled independently at the beginning of each skill segment and remain fixed throughout the segment. The policy $\pi_\omega(a \mid s, g, z)$ receives the full state $s$, while $g$ specifies the target factor. The remaining factors therefore provide contextual information without directly contributing to the skill objective. For each factor $s_i$, we define a representation function $\phi_i : \mathcal{S}_i \to \mathbb{R}^D$, whose output lies in the same $D$-dimensional space as $z$. This allows skill alignment to be measured by the inner product between the representation displacement and $z$. When $g = e_i$, the objective is defined solely on the selected factor $s_i$. The resulting object-centric DSD objective is

$$\begin{aligned} \sup_{\pi, \phi_i} \quad & \mathbb{E}_{p(\tau, g, z)} \left[ \sum_{t=0}^{H-1} \big(\phi_i(s_{i,t+1}) - \phi_i(s_{i,t})\big)^\top z \right] \\ \text{s.t.} \quad & \|\phi_i(s'_i) - \phi_i(s_i)\|_2 \le 1, \qquad \forall (s, s') \in \mathcal{S}_{\text{adj}},\ g = e_i. \end{aligned} \tag{2}$$

The joint distribution factorizes as $p(\tau, g, z) = p(g)p(z)p_{\pi_\omega}(\tau \mid g, z)$. The adjacent-transition constraint bounds the representation displacement of each object over a single MDP transition (Park et al., 2024), thereby preserving the geometric interpretation of $z$ as a direction in the corresponding object representation space.

### 4.2 Object-Level Intrinsic Reward Weighting with $\mathrm{CAI}_i$

Optimizing the object-level objective under only the unit adjacent-edge constraint assigns the same local displacement budget to all target factors. This budget, however, does not account for how strongly the agent's actions influence each factor. To capture such factor-specific action dependence, we learn a factorized dynamics model $p(s' \mid s, a) = \prod_{i=1}^{N} p_i(s'_i \mid s, a)$ and compute CAI from the predictive distribution of each factor. Given the current state $S = s$, the CAI of factor $i$ is defined as the conditional mutual information between the action $A$ and its next state $S'_i$:

$$\begin{aligned} \mathrm{CAI}_i(s) &:= I_q(A; S'_i \mid S = s) \\ &= \mathbb{E}_{a\sim q(\cdot|s)} \left[ D_{\mathrm{KL}}(p_i(\cdot \mid s, a) \,\|\, p_i(\cdot \mid s)) \right] \\ &= \mathbb{E}_{a\sim q(\cdot|s)} \mathbb{E}_{s'_i \sim p_i(\cdot|s,a)} \left[ \log \frac{p_i(s'_i \mid s, a)}{\int p_i(s'_i \mid s, \tilde{a}) q(\tilde{a} \mid s)\, d\tilde{a}} \right]. \end{aligned} \tag{3}$$

Here, $p_i(s'_i \mid s) = \int p_i(s'_i \mid s, \tilde{a}) q(\tilde{a} \mid s)\, d\tilde{a}$ denotes the action-marginalized prediction, where $q$ is the action distribution used to evaluate the conditional mutual information. In practice, we set $q(a \mid s) = \mathcal{U}(\mathcal{A})$ and approximate this quantity using $K$ independently sampled actions, $a^{(1)}, \ldots, a^{(K)} \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\mathcal{A})$, and form the empirical mixture

$$\begin{aligned} \bar{p}_i^{(K)}(s'_i \mid s) &= \frac{1}{K} \sum_{\ell=1}^{K} p_i(s'_i \mid s, a^{(\ell)}), \\ \mathrm{CAI}_i(s) &= \frac{1}{K} \sum_{k=1}^{K} D_{\mathrm{KL}} \Big( p_i(\cdot \mid s, a^{(k)}) \,\Big\|\, \bar{p}_i^{(K)}(\cdot \mid s) \Big). \end{aligned} \tag{4}$$

We also denote this finite-sample estimator by $\mathrm{CAI}_i(s)$. Proposition 1 in Appendix A.1 establishes that $0 \le \mathrm{CAI}_i(s) \le \log K$, satisfying the nonnegativity requirement for $d(s, s')$. The score is large when different sampled actions induce distinct next-state predictions for factor $i$, and approaches zero when these predictions are similar across actions. It therefore provides a nonnegative, state-dependent weight that emphasizes changes only when the target factor is more strongly influenced by the action. Appendix A.2 further analyzes this behavior across no-contact, contact, and grasping regimes.

We use CAI to weight the object-level intrinsic reward of the selected factor. CAI acts only as a reward weight and does not alter the unit adjacent-edge representation constraint introduced in Section 4.1:

$$\begin{aligned} \sup_{\pi, \phi_i} \quad & \mathbb{E}_{p(\tau, g, z)} \left[ \sum_{t=0}^{H-1} \mathrm{CAI}_i(s_t) \big( \phi_i(s_{i,t+1}) - \phi_i(s_{i,t}) \big)^\top z \right] \\ \text{s.t.} \quad & \|\phi_i(s'_i) - \phi_i(s_i)\|_2 \le 1, \qquad \forall (s, s') \in \mathcal{S}_{\mathrm{adj}},\ g = e_i. \end{aligned} \tag{5}$$

### 4.3 Joint Training and Optimization Objective

We jointly train the factorized dynamics model $p$, the object representations $\{\phi_i\}_{i=1}^{N}$, and the skill-conditioned policy $\pi_\omega$. For the selected object $g = e_i$, we define the one-step intrinsic reward as

$$r_{i,t} = \big( \phi_i(s_{i,t+1}) - \phi_i(s_{i,t}) \big)^\top z, \qquad g = e_i. \tag{6}$$

For each sampled transition $(s_t, s_{t+1})$, the slack of the representation constraint is defined as

$$\Delta_i(s_t, s_{t+1}) = 1 - \|\phi_i(s_{i,t+1}) - \phi_i(s_{i,t})\|_2 . \tag{7}$$

Following the dual optimization scheme of DSD (Park et al., 2023), we optimize the object representation and a nonnegative Lagrange multiplier $\lambda$ using the following objectives:

$$J_\phi(\phi_i) = \mathbb{E}_{(s_t, a_t, s_{t+1}, g=e_i, z)\sim\mathcal{D}} \left[ r_{i,t} + \lambda \min(\epsilon, \Delta_i(s_t, s_{t+1})) \right], \tag{8}$$

**Algorithm 1** Object-Centric Skill Discovery with Causal Action Influence

**Require:** Number of epochs $M$, episodes per epoch $E$, and reference actions $K$

Initialize the skill policy $\pi_\omega(a \mid s, g, z)$, factor representations $\{\phi_i\}_{i=1}^N$, factorized dynamics models $\{p_i\}_{i=1}^N$, multiplier $\lambda \geq 0$, and replay buffer $\mathcal{D}$
**for** $m = 1, \ldots, M$ **do**
  **for** $h = 1, \ldots, E$ **do**
    Sample a target factor $g = e_i \sim p(g)$ and a continuous skill $z \sim p(z)$
    Hold $(g, z)$ fixed, collect a skill trajectory with $\pi_\omega(a \mid s, g, z)$, and store it in $\mathcal{D}$
  **end for**
  Sample a minibatch $\mathcal{B} = \{(s_t, a_t, s_{t+1}, g_t, z_t)\}$ from $\mathcal{D}$
  Update the factorized dynamics models using the transitions in $\mathcal{B}$
  Estimate target-factor $\mathrm{CAI}_i(s_t)$ for $\mathcal{B}$ using $K$ reference actions and Eq. 4
  Update only the representations selected by $g_t = e_i$ by maximizing $J_\phi$ in Eq. 8
  Update $\lambda$ by projected gradient ascent on $J_\lambda$ in Eq. 9
  Compute $R_t$ using Eq. 10 and update $\pi_\omega(a \mid s, g, z)$ with SAC
**end for**

$$J_\lambda = -\lambda\, \mathbb{E}_{(s_t, a_t, s_{t+1}, g=e_i, z)\sim\mathcal{D}} \left[\min(\epsilon, \Delta_i(s_t, s_{t+1}))\right], \qquad \lambda \geq 0, \tag{9}$$

where $\epsilon > 0$ caps the positive constraint slack. Constraint violations contribute to increasing $\lambda$ through the dual update, thereby strengthening the constraint. When $\Delta_i \geq \epsilon$, the clipped term contributes no gradient to the representation update.

For policy optimization, we weight the object-level intrinsic reward by the CAI of the target object:

$$R_t = \mathrm{CAI}_i(s_t)\, r_{i,t}, \qquad g = e_i. \tag{10}$$

The skill-conditioned policy $\pi_\omega(a \mid s, g, z)$ is trained using $R_t$ with Soft Actor-Critic (Haarnoja et al., 2018). The complete training procedure is summarized in Algorithm 1.

## 5 Experiments

We evaluate OCUSD along five dimensions: (1) downstream performance in single-object and multi-object environments; (2) object state-space exploration; (3) task relevance of the learned skills; (4) adherence to the selected target object; and (5) the contributions of object-conditioned skill decomposition and CAI-based reward weighting.

### 5.1 Experimental Setup

**Baselines.** We compare OCUSD with two factorized skill discovery methods, DUSDi and SUSD; three distance-maximizing methods, LSD, CSD, and METRA; and the mutual-information-based method DIAYN. Each baseline retains its original skill formulation and intrinsic objective (Hu et al., 2024; Hosseini & Baghshah, 2026; Park et al., 2022; 2023; 2024; Eysenbach et al., 2018), with implementation details provided in Appendix C.

**Environments.** We evaluate object-level skill effectiveness and downstream transfer in three state-based continuous-control environments. Elden Kitchen is a robotic manipulation environment containing a robot and multiple kitchen objects, derived from the multi-entity interaction setting of ELDEN (Wang et al., 2023). Playground is a compositional interaction environment containing a two-dimensional agent and multiple movable geometric objects (Sancaktar et al., 2022). Gym-Fetch (Plappert et al., 2018) contains a single selectable object and serves as a single-object control condition. Detailed state components, object factorizations, and downstream tasks are described in Appendix B.

### 5.2 Downstream Task Performance

We conduct unsupervised skill pretraining separately in Fetch, Elden Kitchen, and Playground. After pretraining, we freeze the low-level skill policy $\pi_\omega(a \mid s, g, z)$ and train only a high-level skill-

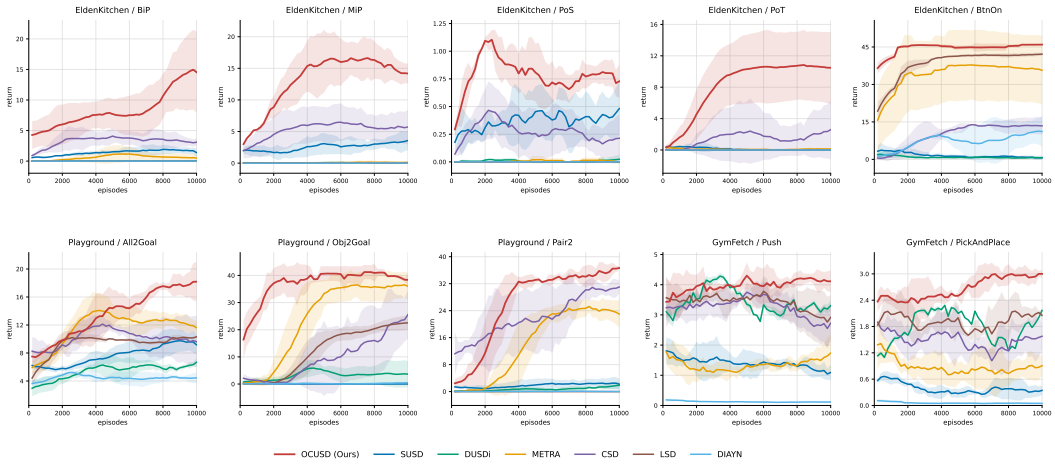



Figure 2: Downstream learning curves on Elden Kitchen, Playground, and GymFetch tasks, averaged over three random seeds.

selection policy on the downstream tasks. Conditioned on the current state and task information, the high-level policy selects $(g, z)$, and the selected skill is executed by the frozen low-level policy for multiple environment steps. This protocol evaluates whether the pretrained skill repertoire provides useful behavioral primitives for downstream control under sparse rewards.

Figure 2 shows that OCUSD achieves the highest final return across all ten downstream tasks, spanning both single-object manipulation and multi-object interaction. The improvement is most pronounced on the more complex Elden Kitchen tasks, while OCUSD also achieves higher final returns and faster learning on Playground and GymFetch. In contrast to the varying performance of the baselines across environments, OCUSD improves over the competing methods across all three domains. Together with the ablation results in Section 5.4, these findings support the role of object conditioning and CAI weighting in improving downstream skill utility.

### 5.3 Skill Effectiveness Analysis

This section directly evaluates the pretrained low-level skills without training an additional high-level policy. We evaluate skill the repertoire from three perspectives: object state-space coverage, task relevance of the induced behaviors, and adherence to the selected object specified by $g$.

#### 5.3.1 Object state-space coverage

For object-interaction tasks, we are interested in whether the learned skills induce changes in object states rather than primarily changing the state of the acting agent. To measure object-state coverage, we randomly resample a skill every 50 environment steps and collect rollouts for 100K steps, or 150K steps in Elden Kitchen. As the number of sampled episodes increases, we cumulatively measure the fraction of discretized bins visited in the state space of each designated object and report the mean across objects.

As shown in Figure 3, OCUSD expands object-state coverage more rapidly and achieves the highest final coverage in all three environments. The improvement across both multi-object domains and Fetch suggests that the learned skills more effectively induce diverse changes in object states, rather than concentrating exploration on the acting agent itself. Appendix F provides detailed coverage curves for each object and for the agent (Figure 7).

#### 5.3.2 Task relevance of learned skills

Broad state-space coverage does not necessarily imply downstream task relevance. To evaluate this directly, we freeze each low-level policy, sample skills from its prior, and score the resulting trajectories using downstream rewards without updating the policy. For each method and seed, we

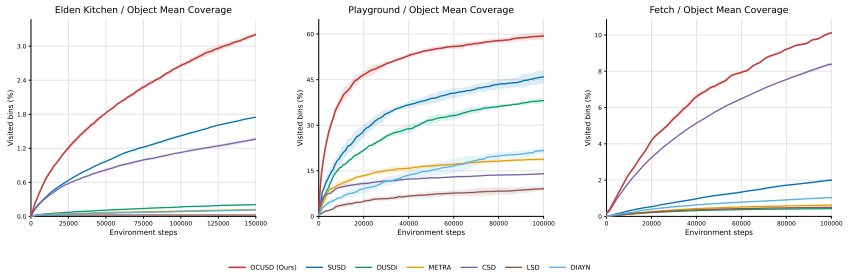



Figure 3: Mean object state-space coverage in Elden Kitchen, Playground, and Fetch. The object mean is computed over all environment objects and excludes the acting agent. Curves show the mean over three random seeds. Detailed coverage results for each object and the agent are provided in Appendix F.1.

collect 10K rollout steps, resampling a skill every 50 environment steps. This measures whether the pretrained repertoire contains task-relevant behaviors such as pushing, grasping, placing, and button manipulation, independently of high-level policy learning. We report Elden Kitchen results in the main text, with Fetch and Playground results in Appendix F.2 (Table 9). We also visualize representative executions of the learned object-conditioned skills in Appendix F.3.

As shown in Table 1, OCUSD achieves the highest return on all five tasks and obtains nonzero reward in every case. The gains are largest on tasks requiring specific object interactions, suggesting that OCUSD learns a repertoire with more task-relevant object-manipulation behaviors.

Table 1: Task returns from randomly sampled skills on five Elden Kitchen tasks. Results are reported as mean ± standard deviation over three seeds; higher is better.

| Task | OCUSD | SUSD | DUSDi | METRA | CSD | LSD | DIAYN |
|---|---|---|---|---|---|---|---|
| BiP | **48.33** ±8.99 | 9.33 ±2.87 | 0.00 ±0.00 | 0.00 ±0.00 | 2.33 ±1.70 | 0.00 ±0.00 | 0.00 ±0.00 |
| MiP | **106.33** ±19.43 | 25.00 ±10.61 | 0.00 ±0.00 | 0.00 ±0.00 | 4.67 ±2.05 | 0.00 ±0.00 | 0.00 ±0.00 |
| PoS | **8.00** ±2.16 | 5.67 ±2.05 | 0.00 ±0.00 | 0.00 ±0.00 | 0.00 ±0.00 | 0.00 ±0.00 | 0.00 ±0.00 |
| PoT | **37.33** ±13.82 | 1.00 ±0.82 | 0.00 ±0.00 | 0.00 ±0.00 | 0.00 ±0.00 | 0.00 ±0.00 | 0.00 ±0.00 |
| BtnOn | **602.67** ±93.88 | 284.00 ±47.59 | 124.67 ±41.12 | 229.00 ±44.86 | 324.33 ±51.72 | 240.33 ±46.45 | 0.00 ±0.00 |

### 5.3.3 Adherence to the object-selection variable $g$

We next evaluate whether the learned behaviors follow the target object specified by $g$. For each target object, we fix $g = e_i$, randomly sample $z$, and execute 100 episodes. We quantify adherence as the fraction of episodes in which the selected object is actually controlled by the agent. Indices $g = e_1, \ldots, e_4$ correspond to pot, butter, meatball, and button in Elden Kitchen, and to cube, light cube, cylinder, and pyramid in Playground, respectively. Table 2 shows consistently high adherence across the evaluated objects, indicating that $g$ provides a reliable target-selection signal rather than merely serving as an additional policy input.

Table 2: Fraction of episodes in which the selected object is controlled, averaged over three seeds.

| Environment | $g = e_1$ | $g = e_2$ | $g = e_3$ | $g = e_4$ |
|---|---|---|---|---|
| Elden Kitchen | 0.99 | 0.97 | 0.89 | 0.96 |
| Playground | 0.90 | 0.99 | 0.94 | 0.98 |

## 5.4 Ablation Study

We isolate the contributions of object-level skill decomposition and CAI reward weighting using three variants trained under the same pretraining budget. **(1) Full OCUSD** uses the object-conditioned skill $\zeta = (g, z)$, the object representation $\phi_i$, the unit adjacent-edge constraint, and the

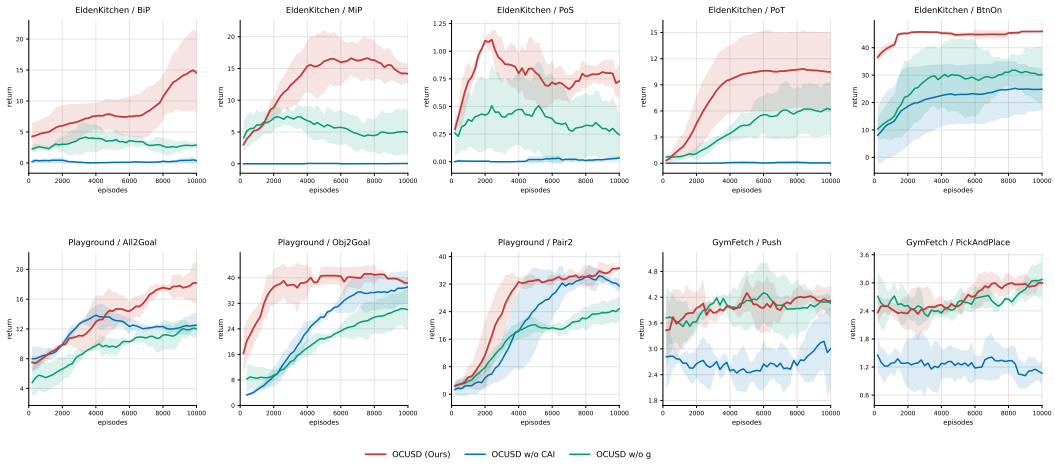



Figure 4: Downstream-task performance of OCUSD and its ablated variants under identical pre-training and evaluation protocols. Curves show the mean over three random seeds.

CAI-weighted reward $R_t = \mathrm{CAI}_i(s_t) r_{i,t}$. **(2) w/o CAI** retains the object-level decomposition and constraint but uses $R_t = r_{i,t}$. **(3) w/o** $g$ removes the object-selection variable and learns a global continuous skill $z$ over the complete state, similarly to holistic baselines. Without a selected target, the last variant defines the transition distance as $d(s, s') = \sum_i \mathrm{CAI}_i(s)$, using the summed influence over all objects as the upper-bound constraint on global representation displacement and its intrinsic reward. Figure 4 compares the downstream performance of these variants under an identical evaluation protocol.

## 6 Conclusion

We introduced an object-centric framework for unsupervised skill discovery to address the ambiguity between global skills and target objects in multi-object environments. By separating object selection from the continuous skill direction, our formulation allows a high-level controller to explicitly specify the target object while the low-level policy learns meaningful skills in an object-level representation space. We further weight the intrinsic reward by object-level causal action influence, emphasizing target-object changes that exhibit stronger action dependence. Across three object-interaction environments, the resulting skills achieve higher object-state coverage, stronger target-object adherence, and better downstream performance than the evaluated baselines.

Our method currently assumes state-based observations with predefined object-level state factors, limiting its direct applicability to environments with unstructured visual observations. Future work will explore integrating object-centric perception to extend the framework to image-based and more complex multi-object environments.

## AI USE STATEMENT

During the preparation of this manuscript, we used generative AI tools for language editing, proofreading, and improving the clarity of technical writing. We did not use generative AI tools to generate experimental results, develop the core methodology, or make scientific claims. All AI-assisted suggestions were reviewed and verified by the authors, who take full responsibility for the final content of this work.

## REPRODUCIBILITY STATEMENT

We provide detailed descriptions of the proposed objective, optimization procedure, and experimental settings throughout the paper and appendix. The object-conditioned skill objective, CAI-based reward weighting, and joint optimization procedure are presented in Sections 4.1–4.3. Environment details and the complete implementation settings, including hyperparameters and evaluation protocols, are provided in Appendices B and E. We report results across multiple random seeds and provide the ablation study in Section 5.4 to analyze the contribution of each component. Source code and supplementary materials will be released to facilitate reproduction of the reported results.

## REFERENCES


Joshua Achiam, Harrison Edwards, Dario Amodei, and Pieter Abbeel. Variational option discovery algorithms. *arXiv preprint arXiv:1807.10299*, 2018.

Peter Battaglia, Razvan Pascanu, Matthew Lai, Danilo Jimenez Rezende, et al. Interaction networks for learning about objects, relations and physics. *Advances in neural information processing systems*, 29, 2016.

Serena Booth, W Bradley Knox, Julie Shah, Scott Niekum, Peter Stone, and Alessandro Allievi. The perils of trial-and-error reward design: misdesign through overfitting and invalid task specifications. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 5920–5929, 2023.

Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. Openai gym. *arXiv preprint arXiv:1606.01540*, 2016.

Víctor Campos, Alexander Trott, Caiming Xiong, Richard Socher, Xavier Giró-i Nieto, and Jordi Torres. Explore, discover and learn: Unsupervised discovery of state-covering skills. In *International conference on machine learning*, pp. 1317–1327. PMLR, 2020.

Jonathan Colaço Carr, Qinyi Sun, and Cameron Allen. Focused skill discovery: Learning to control specific state variables while minimizing side effects. *arXiv preprint arXiv:2510.04901*, 2025.

Jongwook Choi, Sungtae Lee, Xinyu Wang, Sungryull Sohn, and Honglak Lee. Unsupervised object interaction learning with counterfactual dynamics models. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 38, pp. 11570–11578, 2024.

Caleb Chuck, Fan Feng, Carl Qi, Chang Shi, Siddhant Agarwal, Amy Zhang, and Scott Niekum. Null counterfactual factor interactions for goal-conditioned reinforcement learning. In *International Conference on Learning Representations*, volume 2025, pp. 21220–21244, 2025.

Benjamin Eysenbach, Abhishek Gupta, Julian Ibarz, and Sergey Levine. Diversity is all you need: Learning skills without a reward function. *arXiv preprint arXiv:1802.06070*, 2018.

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

Karol Gregor, Danilo Jimenez Rezende, and Daan Wierstra. Variational intrinsic control. *arXiv preprint arXiv:1611.07507*, 2016.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pp. 1861–1870. Pmlr, 2018.

Steven Hansen, Will Dabney, Andre Barreto, Tom Van de Wiele, David Warde-Farley, and Volodymyr Mnih. Fast task inference with variational intrinsic successor features. *arXiv preprint arXiv:1906.05030*, 2019.

Seyed Mohammad Hadi Hosseini and Mahdieh Baghshah. Susd: Structured unsupervised skill discovery through state factorization. In *International Conference on Learning Representations*, volume 2026, pp. 103582–103602, 2026.

Jiaheng Hu, Zizhao Wang, Peter Stone, and Roberto Martín-Martín. Disentangled unsupervised skill discovery for efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 37:76529–76552, 2024.

Paul Jansonnie, Bingbing Wu, Julien Perez, and Jan Peters. Unsupervised skill discovery for robotic manipulation through automatic task generation. In *2024 IEEE-RAS 23rd International Conference on Humanoid Robots (Humanoids)*, pp. 926–933. IEEE, 2024.

Pierre-Alexandre Kamienny, Jean Tarbouriech, Sylvain Lamprier, Alessandro Lazaric, and Ludovic Denoyer. Direct then diffuse: Incremental unsupervised skill discovery for state covering and goal reaching. *arXiv preprint arXiv:2110.14457*, 2021.

Michael Kearns and Daphne Koller. Efficient reinforcement learning in factored mdps. In *IJCAI*, volume 16, pp. 740–747, 1999.

Hyunseung Kim, Byung Kun Lee, Hojoon Lee, Dongyoon Hwang, Sejik Park, Kyushik Min, and Jaegul Choo. Learning to discover skills through guidance. *Advances in Neural Information Processing Systems*, 36:28226–28254, 2023.

Thomas Kipf, Elise Van der Pol, and Max Welling. Contrastive learning of structured world models. *arXiv preprint arXiv:1911.12247*, 2019.

Michael Laskin, Denis Yarats, Hao Liu, Kimin Lee, Albert Zhan, Kevin Lu, Catherine Cang, Lerrel Pinto, and Pieter Abbeel. Urlb: Unsupervised reinforcement learning benchmark. *arXiv preprint arXiv:2110.15191*, 2021.

Michael Laskin, Hao Liu, Xue Bin Peng, Denis Yarats, Aravind Rajeswaran, and Pieter Abbeel. Unsupervised reinforcement learning with contrastive intrinsic control. *Advances in Neural Information Processing Systems*, 35:34478–34491, 2022.

Hao Liu and Pieter Abbeel. Aps: Active pretraining with successor features. In *International Conference on Machine Learning*, pp. 6736–6747. PMLR, 2021.

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.

Shakir Mohamed and Danilo Jimenez Rezende. Variational information maximisation for intrinsically motivated reinforcement learning. *Advances in neural information processing systems*, 28, 2015.

Ofir Nachum, Shixiang Shane Gu, Honglak Lee, and Sergey Levine. Data-efficient hierarchical reinforcement learning. *Advances in neural information processing systems*, 31, 2018.

Sherjil Ozair, Corey Lynch, Yoshua Bengio, Aaron Van den Oord, Sergey Levine, and Pierre Sermanet. Wasserstein dependency measure for representation learning. *Advances in Neural Information Processing Systems*, 32, 2019.

Seohong Park, Jongwook Choi, Jaekyeom Kim, Honglak Lee, and Gunhee Kim. Lipschitz-constrained unsupervised skill discovery. In *International Conference on Learning Representations*, 2022.

Seohong Park, Kimin Lee, Youngwoon Lee, and Pieter Abbeel. Controllability-aware unsupervised skill discovery. *arXiv preprint arXiv:2302.05103*, 2023.

Seohong Park, Oleh Rybkin, and Sergey Levine. Metra: Scalable unsupervised rl with metric-aware abstraction. In *International Conference on Learning Representations*, volume 2024, pp. 18579–18603, 2024.

Silviu Pitis, Elliot Creager, and Animesh Garg. Counterfactual data augmentation using locally factored dynamics. *Advances in Neural Information Processing Systems*, 33:3976–3990, 2020.

Matthias Plappert, Marcin Andrychowicz, Alex Ray, Bob McGrew, Bowen Baker, Glenn Powell, Jonas Schneider, Josh Tobin, Maciek Chociej, Peter Welinder, et al. Multi-goal reinforcement learning: Challenging robotics environments and request for research. *arXiv preprint arXiv:1802.09464*, 2018.

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.

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.

Archit Sharma, Shixiang Gu, Sergey Levine, Vikash Kumar, and Karol Hausman. Dynamics-aware unsupervised discovery of skills. *arXiv preprint arXiv:1907.01657*, 2019.

Richard S Sutton, Doina Precup, and Satinder Singh. Between mdps and semi-mdps: A framework for temporal abstraction in reinforcement learning. *Artificial intelligence*, 112(1-2):181–211, 1999.

Emanuel Todorov, Tom Erez, and Yuval Tassa. Mujoco: A physics engine for model-based control. In *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033. IEEE, 2012.

Zizhao Wang, Xuesu Xiao, Zifan Xu, Yuke Zhu, and Peter Stone. Causal dynamics learning for task-independent state abstraction. *arXiv preprint arXiv:2206.13452*, 2022.

Zizhao Wang, Jiaheng Hu, Peter Stone, and Roberto Martín-Martín. Elden: Exploration via local dependencies. *Advances in Neural Information Processing Systems*, 36:15456–15474, 2023.

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.

David Warde-Farley, Tom Van de Wiele, Tejas Kulkarni, Catalin Ionescu, Steven Hansen, and Volodymyr Mnih. Unsupervised control through non-parametric discriminative rewards. *arXiv preprint arXiv:1811.11359*, 2018.

Nicholas Watters, Loic Matthey, Matko Bosnjak, Christopher P Burgess, and Alexander Lerchner. Cobra: Data-efficient model-based rl through unsupervised object discovery and curiosity-driven exploration. *arXiv preprint arXiv:1905.09275*, 2019.

Rushuai Yang, Chenjia Bai, Hongyi Guo, Siyuan Li, Bin Zhao, Zhen Wang, Peng Liu, and Xuelong Li. Behavior contrastive learning for unsupervised skill discovery. In *International conference on machine learning*, pp. 39183–39204. PMLR, 2023.

# A Properties and Interpretation of Causal Action Influence

## A.1 Boundedness of CAI

**Proposition 1.** *Fix a state $S = s$ and sample $a^{(1)}, \dots, a^{(K)} \overset{\text{i.i.d.}}{\sim} \mathcal{U}(\mathcal{A})$. For each sampled action, let*

$$p_i^{(k)}(s_i') := p_i(s_i' \mid s, a^{(k)}), \qquad \bar{p}_i^{(K)} := \frac{1}{K} \sum_{k=1}^{K} p_i^{(k)}.$$

*Define the empirical CAI estimator as*

$$\widehat{\mathrm{CAI}}_i^{(K)}(s) = \frac{1}{K} \sum_{k=1}^{K} D_{\mathrm{KL}}\left( p_i^{(k)} \,\middle\|\, \bar{p}_i^{(K)} \right).$$

*Then, for every state $s$ and target factor $i$,*

$$0 \le \widehat{\mathrm{CAI}}_i^{(K)}(s) \le \log K.$$

*Proof.* The lower bound follows directly from the nonnegativity of the KL divergence.

For the upper bound, since $\bar{p}_i^{(K)}$ is the uniform mixture of the $K$ component distributions, for every $k$,

$$\bar{p}_i^{(K)}(s_i') = \frac{1}{K} \sum_{\ell=1}^{K} p_i^{(\ell)}(s_i') \ge \frac{1}{K} p_i^{(k)}(s_i').$$

Hence, on the support of $p_i^{(k)}$,

$$\frac{p_i^{(k)}(s_i')}{\bar{p}_i^{(K)}(s_i')} \le K.$$

Therefore,

$$\begin{aligned} D_{\mathrm{KL}}\left( p_i^{(k)} \,\middle\|\, \bar{p}_i^{(K)} \right) &= \mathbb{E}_{s_i' \sim p_i^{(k)}} \left[ \log \frac{p_i^{(k)}(s_i')}{\bar{p}_i^{(K)}(s_i')} \right] \\ &\le \log K. \end{aligned}$$

Averaging over $k = 1, \dots, K$ gives

$$\widehat{\mathrm{CAI}}_i^{(K)}(s) \le \log K,$$

which completes the proof. □

This bound applies specifically to the finite-sample estimator induced by the uniform mixture over the $K$ sampled actions. It should not be interpreted as a universal upper bound on the conditional mutual information associated with the original continuous action distribution. For notational simplicity, we omit the hat and superscript in the main text and continue to denote this empirical estimator by $\mathrm{CAI}_i(s)$.

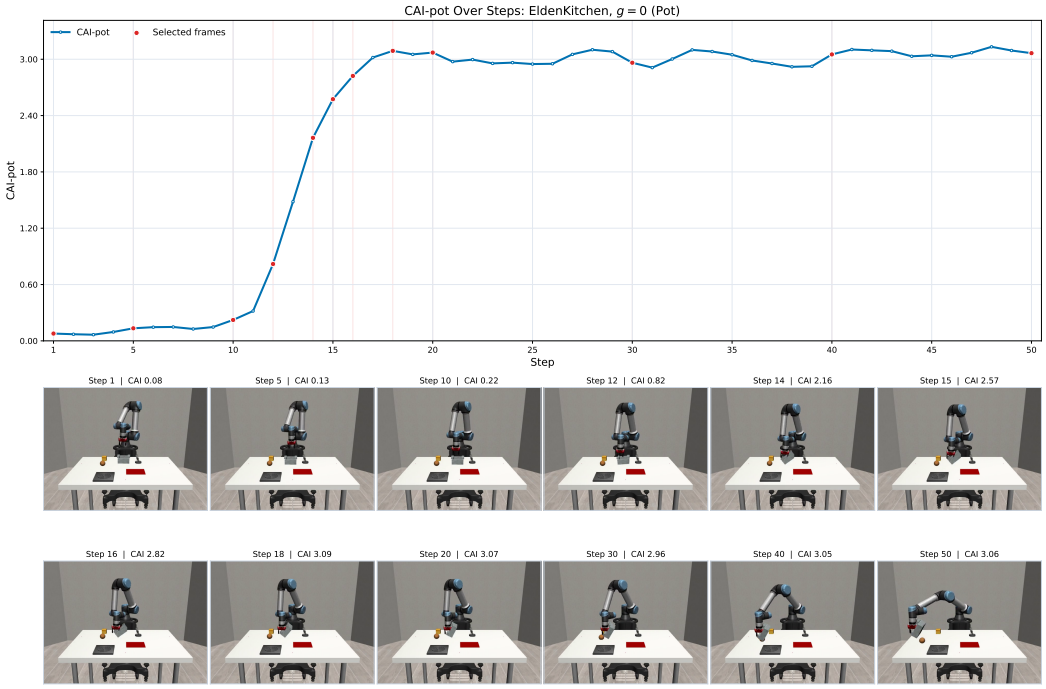



Figure 5: CAI of the pot object along a representative Elden Kitchen trajectory. Selected frames illustrate the transition from low-influence states to sustained object interaction.

### A.2 CAI as a Measure of Object Controllability

Figure 5 tracks the estimated CAI of the pot along a representative Elden Kitchen trajectory. CAI remains near zero while the robot has little immediate influence on the pot, rises sharply as interaction is established, and stabilizes around 3.1 during manipulation. With $K = 32$ reference actions, this value is close to the finite-sample upper bound $\log 32 \approx 3.47$ from Proposition 1. The progression shows that CAI distinguishes different levels of object controllability: as a state-dependent reward weight, it suppresses intrinsic rewards when the target cannot be influenced and amplifies them once the target becomes controllable, thereby promoting meaningful object-level skills.

**Stability with respect to the number of reference actions.** We evaluate the stability of CAI across different numbers of sampled reference actions $K$. For each $K$, we evaluate the corresponding converged policy over 100 episodes from grasping states and report the estimated CAI in Table 3. The CAI estimates show low variability across episodes, with coefficients of variation of $0.0402$, $0.0494$, $0.0505$, and $0.0626$, respectively. Since $0 \leq \mathrm{CAI}_K(s) \leq \log K$ (Proposition 1), the raw CAI scale increases with $K$. After normalization by $\log K$, the scores remain within $[0.861, 0.899]$, indicating stable CAI estimates across different $K$.

Table 3: CAI estimation stability across different numbers of sampled reference actions $K_s$. CAI scores are reported as mean $\pm$ standard deviation over 100 evaluation episodes from grasping states.

| Metric | $K = 8$ | $K = 16$ | $K = 32$ | $K = 64$ |
|---|---|---|---|---|
| CAI score | $1.789 \pm 0.072$ | $2.451 \pm 0.121$ | $3.106 \pm 0.157$ | $3.738 \pm 0.234$ |
| $\log K$ | 2.079 | 2.773 | 3.466 | 4.159 |
| CAI/ $\log K$ | 0.861 | 0.884 | 0.896 | 0.899 |

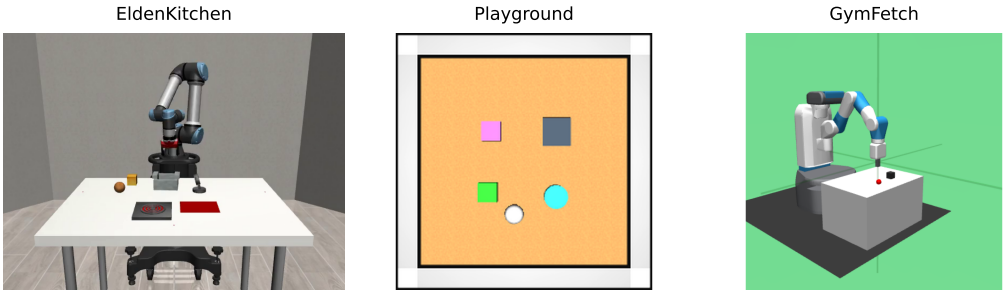



Figure 6: The three evaluation environments: Elden Kitchen, Playground, and GymFetch.

## B Environment Details

**General setup.** All environments provide state observations and continuous primitive actions. During unsupervised pretraining, a sampled skill is executed for a 50-step episode. The acting robot or planar agent remains part of the policy observation but is not selectable by $g$. For downstream evaluation, we freeze the pretrained low-level policy and train a high-level SAC policy. Each high-level action selects a skill $(g, z)$ for five primitive steps; a 50-decision episode therefore contains at most 250 primitive steps. The corresponding high-level reward is the average reward over those five steps. Both low- and high-level policies use Soft Actor-Critic (Haarnoja et al., 2018). Figure 6 shows the three environments, and Table 4 summarizes their pretraining observations and action spaces.

Table 4: Evaluation environments.

| Environment | Scene | Multi-object | Observation dim. | Action dim. |
|---|---|---|---|---|
| Elden Kitchen | 3D | Yes | 128 | 4 |
| Playground | 2D | Yes | 40 | 2 |
| Fetch | 3D | No | 25 | 4 |

### B.1 Elden Kitchen

Elden Kitchen is a three-dimensional manipulation environment based on the ELDEN benchmark (Wang et al., 2023), containing a robot, pot, butter, meatball, button, and stove. We treat the pot, butter, meatball, and button as selectable object factors, while the robot and stove provide context. Its pretraining state comprises the agent (32), pot (22), butter (18), meatball (19), button (23), and stove (14).

**Downstream tasks.** Downstream observations append 14 task-specific goal dimensions to the 128-dimensional pretraining state, yielding 142 dimensions. Each primitive step receives a binary task-completion reward. The five tasks are **BiP**, placing the butter inside the pot; **MiP**, placing the meatball inside the pot; **PoS**, moving the pot onto the stove; **PoT**, moving the pot into a target region; and **BtnOn**, moving the button joint to its activated position.

### B.2 Playground

Playground is a two-dimensional object-interaction environment introduced by Sancaktar et al. (2022), containing a mobile agent and four movable objects: a light cube (cube_light0), a heavy cube (cube0), a cylinder (cylinder0), and a pyramid (pyramid0); all four are selectable by $g$. The pretraining state contains four agent dimensions and nine dimensions for each object.

**Downstream tasks.** Task-specific information augments the 40-dimensional state, yielding a 62-dimensional high-level observation. We evaluate three tasks. **All2Goal** requires all objects to lie

within $0.45$ of a shared target and awards $0.25$ for each successfully placed object. **Obj2Goal** requires a designated object to reach a target within a tolerance equal to its radius plus $0.05$. **Pair2** requires two designated objects to be separated by at most the sum of their collision radii plus $0.05$. The latter two tasks use binary success rewards.

### B.3 FETCH

Fetch is a three-dimensional robotic manipulation environment based on the Fetch tasks (Plappert et al., 2018), which were released through OpenAI Gym (Brockman et al., 2016) and use MuJoCo physics (Todorov et al., 2012). Its single object is the only factor selectable by $g$, providing a three-dimensional single-object evaluation. The pretraining state contains 10 agent dimensions and 15 object dimensions.

**Downstream tasks.** Downstream observations append a three-dimensional goal coordinate to this 25-dimensional state, yielding 28 dimensions. **Fetch Push** moves the object across the tabletop to a target position, whereas **Fetch Pick-and-Place** grasps and moves it to a three-dimensional target.

## C BASELINE DETAILS

**DUSDi** represents a skill as multiple categorical components aligned with predefined state factors. It maximizes the mutual information between each component and its corresponding factor, suppresses dependence on other factors, and factorizes the Q-function for credit assignment across intrinsic rewards (Hu et al., 2024). **SUSD** assigns an independent continuous skill and representation to every factor, sums object-level intrinsic rewards, and uses dynamics-based negative log-likelihood weights to emphasize underexplored or unpredictable transitions (Hosseini & Baghshah, 2026). Its factor components are executed jointly, producing a compositional global skill. **LSD** samples a continuous skill direction and aligns it with displacement in a global representation, whose scale is constrained by Euclidean distance in the original state space (Park et al., 2022). **CSD** replaces this fixed distance with a policy-dependent measure derived from transition negative log-likelihood, assigning larger distances to transitions that are difficult for the current skill repertoire (Park et al., 2023). Both methods operate on the complete state without identifying the object responsible for the change. **METRA** learns a metric-aware global representation using a Wasserstein dependence objective and temporal-distance constraint, with continuous skills specifying latent directions (Park et al., 2024). **DIAYN** instead samples discrete skills and trains a discriminator to infer them from visited states; the discriminator reward and policy entropy encourage distinguishable state distributions (Eysenbach et al., 2018). Unlike OCUSD, none of these baselines jointly supplies an explicit single-object selector and an object-level action-influence signal.

## D ADDITIONAL DISCUSSION

DUSDi factorizes skills across state factors and promotes factor-level diversity through mutual information, but it neither provides a single-object selection interface nor explicitly measures action-conditioned controllability (Hu et al., 2024). SUSD adaptively reweights object-level intrinsic rewards to emphasize underexplored or difficult-to-control factors, yet its jointly executed factor components still form a global compositional skill rather than a single-target skill (Hosseini & Baghshah, 2026). SkiLD models local causal interactions, but its reported three-dimensional setting assumes a ground-truth causal graph, introducing supervision unavailable in our environments and preventing a directly controlled comparison (Wang et al., 2024). In contrast, OCUSD explicitly selects a target object with $g$ and weights its intrinsic reward using object-level CAI.

## E IMPLEMENTATION DETAILS

This appendix summarizes the implementation details not specified in Appendix B. Unless stated otherwise, experiments use three independent seeds, and results report their mean and standard deviation.

### E.1 Unified Comparison Setup

All methods use the same environment-interaction budget, skill duration, and downstream hierarchical-control protocol. DUSDi and SUSD receive the same state factorization used by OCUSD, whereas LSD, CSD, METRA, and DIAYN receive the complete state without a target-object mask or object-only oracle input. Each method preserves its original skill type, intrinsic objective, and main optimization mechanism after adapting the observation, action, and skill dimensions. The downstream stage always loads the matching pretrained state ordering and skill interface, freezes the low-level policy, and trains only the high-level task policy.

### E.2 Skill and Dynamics Interfaces

OCUSD samples $g$ uniformly from the active objects at the beginning of each pretraining episode. It samples $z$ from a standard Gaussian and normalizes it to unit length; both variables remain fixed for the 50-step skill trajectory. The policy directly receives $[s, g, z]$ without an additional fusion module. Table 5 lists the resulting dimensions.

For all baselines, we follow the skill parameterizations and training objectives used in their original papers, including the corresponding intrinsic rewards and discriminator or representation objectives.

Table 5: OCUSD skill dimensions.

| Environment | Object choices $N$ | $\dim(z)$ | $\dim(g, z)$ |
|---|---|---|---|
| Elden Kitchen | 4 | 3 | 7 |
| Fetch | 1 | 3 | 4 |
| Playground | 4 | 3 | 7 |

### E.3 Unsupervised Pretraining

Table 6 gives the main OCUSD optimization settings. OCUSD begins gradient updates after the replay buffer contains $50{,}000$ transitions and saves evaluation checkpoints every 4,000 epochs. All methods are trained for the same interaction budget.

Table 6: Hyperparameters for unsupervised pretraining.

| Hyperparameter | Value |
|---|---|
| Optimizer | Adam |
| Learning rate | $10^{-4}$ |
| Discount factor $\gamma$ | $0.99$ |
| Skill $z$ unit length | True |
| Episodes per epoch | 8 |
| Trajectory length | 50 |
| Minibatch size | 256 |
| Gradient steps per epoch | 100 |
| Initial lambda | 3000 |
| Initial entropy coefficient | $0.1$ (adaptive) |
| Replay-buffer capacity | $2 \times 10^6$ |
| Policy/encoder MLP | $2 \times 512$ |
| MLP activation | ReLU |
| Training epochs | 20,000 |
| $K$ (reference actions for CAI estimation) | 32 |

For baselines, method-specific quantities such as SUSD's factor weights, DIAYN's discriminator, and the distance objectives of LSD, CSD, and METRA follow their original implementations.

### E.4 Downstream Hierarchical Training

The pretrained low-level policy is frozen. At each decision, the high-level policy selects a skill that is executed for five primitive steps. The remaining settings are summarized in Table 7.

Table 7: Hyperparameters for downstream high-level training.

| Hyperparameter | Value |
|---|---|
| Optimizer | Adam |
| Learning rate | $10^{-4}$ |
| Discount factor $\gamma$ | 0.99 |
| Skill $z$ unit length | True |
| Episodes per epoch | 1 |
| High-level horizon | 50 |
| Skill repeat | 5 primitive steps |
| Gradient steps per epoch | 50 |
| Replay-buffer capacity | $10^6$ |
| Training epochs | 10,000 |
| Target smoothing | 0.995 ($\tau = 0.005$) |
| Initial entropy coefficient | 0.1 (adaptive) |
| Evaluation frequency | Every 200 epochs |
| Evaluation episodes | 100 |

### E.5 Training Control and Reporting

Random seeds are fixed for each run. Results are aligned on a common evaluation grid, with final performance reported at the last common evaluation point. Shaded regions and tabulated deviations denote the standard deviation across seeds. Curve smoothing is used only for visualization.

### E.6 Experimental Setup

The experiments are performed on a workstation with the hardware and system configurations listed in Table 8. All models are trained with GPU acceleration.

Table 8: Hardware and system configurations.

| Item | Details |
|---|---|
| Operating System | Ubuntu 18.04.2 LTS |
| CUDA Version | 12.4 |
| CPU | 2 $\times$ 64 Cores (128 Threads) |
| GPU | NVIDIA GeForce RTX 3090 (24 GB) |

## F Additional Experimental Results

### F.1 Per-Object State-Space Coverage

Figure 7 provides the detailed coverage curves for each object in Elden Kitchen, Playground, and Fetch, complementing the object-averaged results in the main text. These curves show how broadly each method explores the discretized state space of individual objects and reveal whether aggregate improvements are shared across objects or dominated by a small subset. The curve labeled *Entity Mean* averages coverage over all state entities in an environment, including both the acting agent and the objects; it therefore differs from the object-only mean reported in the main text.

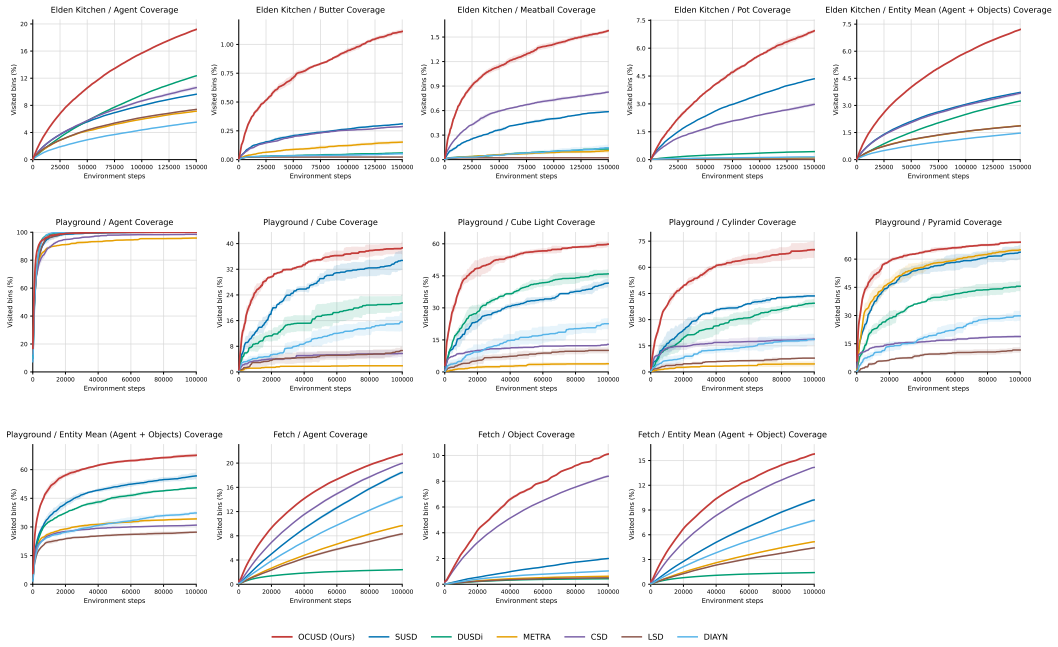



Figure 7: Per-entity state-space coverage in Elden Kitchen, Playground, and Fetch. *Entity Mean* includes the agent and all objects, whereas the main-text metric averages only over controllable objects.

### F.2 Additional Task-Relevance Results

Table 9 reports the task returns obtained by randomly executing pretrained skills in Playground and Fetch, following the same protocol as the Elden Kitchen evaluation in the main text.

Table 9: Task returns from randomly sampled skills in Playground and Fetch. Results are mean ± standard deviation over three seeds; higher is better.

| Playground | OCUSD | SUSD | DUSDi | METRA | CSD | LSD | DIAYN |
|---|---|---|---|---|---|---|---|
| All2Goal | **246.00** ±4.83 | 236.08 ±4.60 | 211.08 ±21.35 | 229.08 ±11.27 | 231.67 ±11.65 | 217.25 ±16.61 | 209.08 ±20.00 |
| Obj2Goal | **225.33** ±41.91 | 180.67 ±24.85 | 150.00 ±75.61 | 189.67 ±18.19 | 180.00 ±70.41 | 156.00 ±48.36 | 186.00 ±84.02 |
| Pair2 | **162.00** ±37.07 | 52.00 ±23.85 | 11.67 ±8.73 | 0.00 ±0.00 | 99.67 ±31.33 | 0.00 ±0.00 | 0.00 ±0.00 |

| Fetch | OCUSD | SUSD | DUSDi | METRA | CSD | LSD | DIAYN |
|---|---|---|---|---|---|---|---|
| Push | **721.67** ±50.63 | 652.67 ±62.26 | 583.00 ±170.42 | 558.67 ±28.99 | 648.00 ±84.11 | 607.00 ±67.21 | 510.67 ±64.71 |
| PickAndPlace | **481.00** ±79.65 | 219.33 ±60.54 | 253.00 ±55.54 | 159.67 ±73.33 | 278.33 ±52.00 | 221.33 ±99.59 | 160.00 ±19.20 |

### F.3 Qualitative Object-Conditioned Skill Executions

For clarity, we visualize representative skill trajectories for a subset of the selectable objects across the three environments. In each row of Figure 8, we fix $g$ to the indicated target object, randomly sample a unit direction $z$ from the skill prior, and hold $(g, z)$ fixed throughout the trajectory. The resulting sequences illustrate that the learned skills follow the selected object while producing diverse object-specific changes determined by $z$.

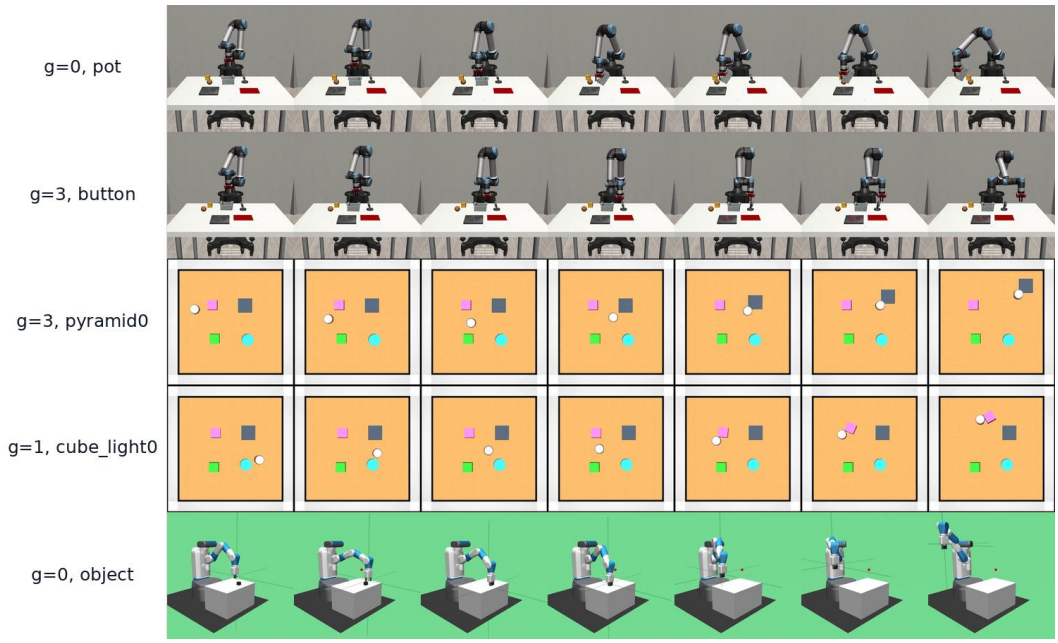



Figure 8: Representative object-conditioned skill executions. Each row fixes $g$ to the labeled target object and uses a randomly sampled $z$; frames progress from left to right.