# RICH OUTCOMES, SELECTIVE CONSTRAINTS: GOAL OVERSPECIFICATION IN OFFLINE GOAL-CONDITIONED LEARNING

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

## ABSTRACT

Reward-free offline goal-conditioned learning obtains generality by treating achieved outcomes as reusable supervision, enabling policies to learn from offline experience without task-specific rewards. Yet a complete achieved outcome can contain more information than a downstream task requires. We show that conflating what an outcome describes with what a task demands creates *goal overspecification*: full-state hindsight objectives can require reproducing properties that the current success criterion leaves unconstrained. This tension is particularly important for general-purpose learning, where such properties should be retained because they may define other tasks. We therefore separate outcome information from task constraints with Goal-Set Hindsight Relabeling (GS-HER). GS-HER pairs a reference outcome with a query that defines a task-dependent equivalence class and makes the goal-conditioning representation invariant to unqueried outcome properties, while preserving the base learner's hindsight sampling, supervision, target construction, and optimization objective. Across five heterogeneous offline goal-conditioned learners and OGBench manipulation tasks, replacing full-state semantics with the oracle task projection yields higher mean success in all $25/25$ backbone–task pairs; GS-HER recovers much of this gap, increasing average success from $24.4\%$ to $56.5\%$, compared with $60.0\%$ for the oracle projection. In contrast, gains largely disappear on PointMaze, where outcome and task semantics are already aligned. A single GS-HER checkpoint can additionally support multiple object- and robot-centric success predicates from the query family represented during training by changing only the inference-time query. Our results suggest a simple principle for general-purpose robot learning: retain outcome information broadly, but impose task constraints selectively.

## 1 INTRODUCTION

Learning general-purpose robotic behavior from offline, reward-free experience offers an appealing route toward scalable robot learning. Offline goal-conditioned learning provides a particularly simple recipe: learn to reach desired outcomes from previously collected trajectories, using hindsight relabeling to turn outcomes achieved later in a trajectory into supervision for earlier states (Andrychowicz et al., 2017). Because this supervision comes from the data itself, a single dataset can support many goal-reaching problems without requiring task-specific rewards or demonstrations. This has made reward-free offline goal-conditioned learning a promising framework for reusable behavioral pretraining (Park et al., 2025).

A common way to preserve this generality is to use complete achieved states as goals. Doing so is task-agnostic and retains potentially useful information: an achieved robot state may simultaneously encode object pose, robot configuration, gripper state, velocities, and other properties whose relevance to future tasks is unknown during pretraining. However, this formulation hides an important tension: a complete description of an achieved outcome is not necessarily a complete specification of what must be reproduced.

Consider a trajectory in which a robot places a cube at a desired position while ending in a particular arm configuration, gripper state, and cube orientation. The achieved state records all of these prop-



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.

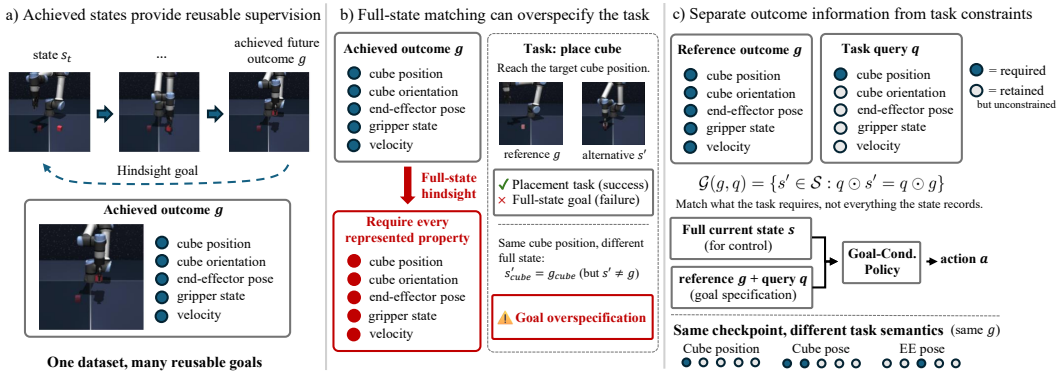


Figure 1: **Goal overspecification from complete achieved outcomes.** (a) Reward-free goal-conditioned learning reuses achieved future outcomes as supervision, motivating rich outcome representations whose future relevance is unknown. (b) Full-state hindsight treats every represented property as a requirement: an alternative outcome can solve the downstream task while failing to reproduce the complete reference state. (c) GS-HER separates these roles: the reference outcome $g$ retains what was achieved, while a query $q$ specifies which outcome distinctions define the queried goal. The same retained information can therefore support different task-dependent constraints.

erties because they occurred, but a downstream placement task may require only the cube position. Under full-state goal conditioning, two outcomes with the same successful cube position but different robot configurations, gripper states, velocities, or orientations remain distinct goals. The learner is therefore asked to preserve distinctions that the downstream task itself does not require. We refer to this mismatch as *goal overspecification*. Related concerns have been observed for expressive goal representations such as images, which may encode more scene information than the intended behavior requires (Nair et al., 2022). Recent goal-representation methods similarly seek representations that suppress exogenous goal variation while preserving information sufficient for goal-reaching control (Park et al., 2026). We study a complementary axis: rather than learning a single invariant representation of an outcome, we keep the represented outcome rich and study which of its retained properties should constitute requirements for a particular task. A property that is unconstrained by one task may still define another and therefore should not necessarily be removed from the outcome representation.

Here, we isolate this distinction in offline goal-conditioned learning, where achieved outcomes themselves provide supervision. A reusable learner should retain outcome information whose future relevance is unknown, while any individual task should constrain only the properties that define its goal. Simply discarding variables globally is therefore insufficient: a property irrelevant to placement may define an alignment task, and state information irrelevant to terminal success may still be essential for control. This motivates our central principle: *retain outcome information broadly, but impose task constraints selectively*.

We study this principle in factored state spaces, where retained outcome information and task requirements can be separated explicitly, and instantiate it with Goal-Set Hindsight Relabeling (GS-HER).

GS-HER separates a reference outcome $g$ from a binary query $q \in \{0, 1\}^D$ specifying which outcome properties distinguish the desired goal. The query induces the equivalence class

$$\mathcal{G}(g, q) = \{s' \in \mathcal{S} : q \odot s' = q \odot g\}. \tag{1}$$

A future state $g = s_{t+k}$ achieved in hindsight can therefore serve as a witness for every query-defined equivalence class that contains it: the demonstrated trajectory reaches that class within $k$ steps.

GS-HER implements this distinction at the goal-conditioning interface. For a sampled outcome, it preserves the base learner's reference-outcome sampling, learner-specific supervision, target construction, and optimization objective, while making the goal representation invariant to the unqueried completion of $g$. The complete current state remains available for control. This controlled

intervention lets us test whether changing goal semantics alone can recover the cost of goal over-specification.

We evaluate the goal-overspecification hypothesis on state-based OGBench across manipulation and navigation domains and five heterogeneous offline goal-conditioned learners. In manipulation, replacing full-state semantics with the oracle task-specific projection improves performance in all $25/25$ backbone–task pairs. GS-HER recovers much of this gap without fixing the task projection before training, increasing average manipulation success from $24.4\%$ to $56.5\%$, compared with $60.0\%$ for the oracle projection. Conversely, on PointMaze, where represented state and downstream task semantics are already aligned, the oracle-projection gap becomes negligible on average. A single GS-HER checkpoint can additionally expose multiple object- and robot-centric success predicates by changing only the inference-time query.

**Contributions.** We make three contributions: (i) we isolate and empirically characterize *goal overspecification* in offline goal-conditioned learning, showing that complete outcome representations can distinguish states that the downstream task treats as equivalent; (ii) we formalize task-dependent goal equivalence classes and instantiate them with GS-HER, a query-conditioned goal interface that preserves the base learner's hindsight supervision and learning objective; and (iii) across five heterogeneous offline goal-conditioned learners, we show that this representation-level intervention recovers substantial performance when full-state and downstream task semantics are misaligned, provides little benefit when they are already aligned, and allows one trained checkpoint to expose multiple queried goal specifications.

## 2 Related Work

**Offline goal-conditioned learning and hindsight.** Goal-conditioned reinforcement learning learns policies or value functions conditioned on desired outcomes (Schaul et al., 2015), while hindsight relabeling reuses achieved future states as goals to provide self-supervision under sparse rewards (Andrychowicz et al., 2017). This idea has become a central ingredient of reward-free and offline goal-conditioned learning, where a fixed dataset can support many downstream goals without task-specific reward annotation (Eysenbach et al., 2020; Park et al., 2025). Recent offline GCRL methods instantiate this principle through behavioral cloning, value learning, contrastive objectives, and hierarchical control (Park et al., 2025). We study a semantic choice shared by these approaches when achieved states are reused as goals: whether all information recorded by an outcome should also become a requirement for reproducing it.

**Partial goals, structured goals, and task relabeling.** Goal masking has been used to deactivate goal coordinates and construct curricula of varying difficulty (Eppe et al., 2018), while Multi-Criteria HER independently relabels components of structured goals to generate partially achieved training targets (Lanier, 2019). Generalized Hindsight and HIPI similarly reinterpret experience under alternative tasks or reward functions (Li et al., 2020; Eysenbach et al., 2020). These works provide established mechanisms for selective goal specification and task relabeling. Most closely related, Ståhlberg & Geffner (2026) show that goal-conditioned learning can benefit from replacing complete relational states with subsets or lifted sets of atoms. We study how the same underlying issue manifests in offline continuous-control GCRL with fixed factored states, and use GS-HER to test whether task-irrelevant distinctions retained by complete outcomes form a systematic bottleneck.

**Goal representations and task specification.** A complementary line of work studies what information a goal representation should preserve. Visual and latent goal-conditioned RL learns compact representations for broad self-supervised skill acquisition (Nair et al., 2018; Pong et al., 2019), while language-conditioned rewards and policies provide more abstract interfaces for specifying intended behavior (Fu et al., 2019; Nair et al., 2022; Myers et al., 2023; Ma et al., 2023). Prior work has also observed that goal images can over-specify desired behavior by encoding scene properties beyond the intended task (Nair et al., 2022).

More recently, Dual Goal Representations (Park et al., 2026) encodes goals through their temporal relations to other states, yielding representations that depend on the environment dynamics rather than the original state encoding and can filter exogenous variation while retaining sufficient information for optimal goal reaching. This line of work focuses on which distinctions a goal representation

should preserve. Our focus is complementary: given a rich retained outcome, which of those distinctions should a particular task require? This distinction matters when relevance is task-dependent, because a property unconstrained by one task may define another.

We examine this question in factored state spaces, where retained outcome information and task-required constraints can be separated and measured explicitly.

# 3 GOAL-SET HINDSIGHT RELABELING

We now formalize the distinction introduced in Sec. 1 between describing an achieved outcome and specifying which distinctions matter for a task.

GS-HER isolates task-selective goal semantics at the representation level. It preserves the base learner's reference-outcome sampling and target construction, and changes only the representation through which a sampled outcome is presented to the goal-conditioned learner.

## 3.1 FROM ACHIEVED OUTCOMES TO QUERY-DEFINED EQUIVALENCE CLASSES

We consider an offline dataset $\mathcal{D}$ of transitions with states $s \in \mathcal{S} \subseteq \mathbb{R}^D$ and actions $a \in \mathcal{A}$. As in standard hindsight-based offline GCRL, the underlying learner samples reference outcomes $g$ from states in the dataset.

A nonempty binary query $q \in \{0,1\}^D$ specifies which coordinates distinguish the desired outcome and induces

$$s' \equiv_q g \iff q \odot s' = q \odot g, \qquad \mathcal{G}(g,q) = [g]_q = \{s' : q \odot s' = q \odot g\}. \tag{2}$$

For $q = \mathbf{1}$ this recovers full-state goal semantics; partial queries identify outcomes that differ only in inactive coordinates.

This interpretation is directly compatible with hindsight supervision. If the base relabeler chooses $g = s_{t+k}$, then $g \in \mathcal{G}(g,q)$; the demonstrated suffix therefore certifies $\tau_{\mathcal{G}(g,q)}(s_t) \leq k$, where $\tau_{\mathcal{G}}$ denotes the first hitting time of the set along that suffix. Thus, $k$ is a sound upper bound, not an assertion that the class is first reached exactly at $t + k$. GS-HER does not search for earlier query-equivalent states, so this witness can be loose.

Goal overspecification occurs when the downstream task identifies outcomes that full-state conditioning still distinguishes: $s_1 \equiv_{q^\star} s_2$ while $s_1 \neq s_2$. The issue is therefore not missing state information, but retaining task-irrelevant distinctions in the goal representation.

## 3.2 TASK-DEPENDENT GOAL INVARIANCE

Providing $q$ alongside the complete goal is insufficient because a function approximator could still exploit inactive goal values. We therefore construct

$$\psi_{\mathrm{GS}}(s,g,q) = [q \odot s + (1-q) \odot e_s,\ q \odot g + (1-q) \odot e_g,\ q], \tag{3}$$

where $e_s, e_g \in \mathbb{R}^D$ are learned inactive-coordinate embeddings. Consequently, $q \odot g = q \odot g'$ implies $\psi_{\mathrm{GS}}(s,g,q) = \psi_{\mathrm{GS}}(s,g',q)$.

This invariance applies only to the goal-comparison channel. The complete current state remains available for control through $\pi(a \mid s, \psi_{\mathrm{GS}})$, $V(s, \psi_{\mathrm{GS}})$, or $Q(s,a,\psi_{\mathrm{GS}})$. GS-HER therefore removes task-irrelevant goal distinctions without discarding control-relevant state information.

## 3.3 GS-HER AS A GOAL-CONDITIONING WRAPPER

Let $\mathcal{A}$ denote a base goal-conditioned learner with relabeler $\mathcal{R}_{\mathcal{A}}$. For each sampled transition, the original procedure produces a reference outcome and any learner-specific metadata, $(g,\xi) \sim \mathcal{R}_{\mathcal{A}}(\cdot \mid \tau, t, \mathcal{D})$. For future-trajectory goals in the value-based backbones, $\xi$ includes the sampled temporal offset $k_{\mathrm{traj}}$. GS-HER leaves $(g,\xi)$ unchanged, independently samples $q \sim p_Q(q)$, and replaces the conventional goal input with $\psi_{\mathrm{GS}}$.

If $\mathcal{L}_{\mathcal{A}}$ denotes the original learner objective, the resulting update is abstractly

$$\mathcal{L}_{\mathrm{GS}} = \mathcal{L}_{\mathcal{A}}(s_t, a_t, s_{t+1}, \psi_{\mathrm{GS}}(s_t, g, q), \psi_{\mathrm{GS}}(s_{t+1}, g, q), \xi)\,, \tag{4}$$

with unused arguments omitted as appropriate.

GS-HER preserves $\xi$ and each learner's native supervision: GCIVL, GCIQL, and GCDL retain their original one-sided demonstration-witness constraints derived from $k_{\mathrm{traj}}$, while GCBC and CRL retain their native supervision. Query conditioning therefore changes goal semantics without altering learner-specific temporal supervision or target construction.

The query distribution $p_Q$ determines the represented goal family. Our annotation-free Blockwise variant mixes full-state, random-coordinate, and contiguous-block queries, exploiting locality in the state-vector ordering; the Semantic variant uses privileged physical factor groups (Appendix D).

At inference, changing $q^\star$ in $\pi(a \mid s, \psi_{\mathrm{GS}}(s, g^\star, q^\star))$ changes which distinctions are imposed by the goal without changing the model parameters.

## 4 Experiments: Diagnosing Goal Overspecification

Our experiments are organized around the goal-overspecification hypothesis. Rather than asking only whether GS-HER improves benchmark performance, we use the existing OGBench task structure and oracle task projections to ask:

1. Does full-state conditioning become harmful when the represented outcome contains properties beyond those required by the downstream task?
2. Does this effect persist across qualitatively different goal-conditioned learning algorithms?
3. Can a single model retain broad outcome information while changing which properties define success at inference time?

### 4.1 Experimental Setup

**Environments.** We evaluate on state-based OGBench (Park et al., 2025). Our manipulation suite contains `cube-single-play`, `cube-single-noisy`, `cube-double-play`, `cube-double-noisy`, and `scene-play`. These domains are particularly informative for our hypothesis because their complete observations contain robot configuration, gripper state, velocities, and other properties beyond those used by the official task-success predicates. We additionally evaluate `pointmaze-medium`, `pointmaze-large`, and `antmaze-medium`. PointMaze serves as a control because its state is already aligned with its navigation goal.

**Goal-conditioning variants.** We compare three main conditions. HER-Full conditions the learner on the complete reference outcome, so different full-state completions remain distinct goals. HER-Task is an oracle diagnostic that applies the fixed projection corresponding to the official benchmark task before constructing the goal-conditioning input. It removes distinctions outside the known task projection, but commits to one goal specification before training. GS-HER instead pairs the same reference outcome with a sampled query and makes the goal-conditioning interface invariant to inactive coordinates. Our main Blockwise variant samples full-state, coordinate, and contiguous-block queries without semantic task annotations; the Semantic variant uses privileged physical factor groupings. More details in Appendix D.2.

**Base learners and protocol.** We evaluate GCBC, CRL, GCIVL, GCIQL, and GCDL, covering imitation-, contrastive-, value-, and distance-based GCRL. Within each backbone, all variants share the dataset, transition/reference sampling, learner-specific target construction, architecture depth and width, optimization, and training budget; only the goal-conditioning representation changes, apart from GS-HER's small input-induced parameter increase. All methods use five seeds and the official OGBench success metric. We use a common compute schedule of 200k updates with batch size 8192; full details are in Appendix G.

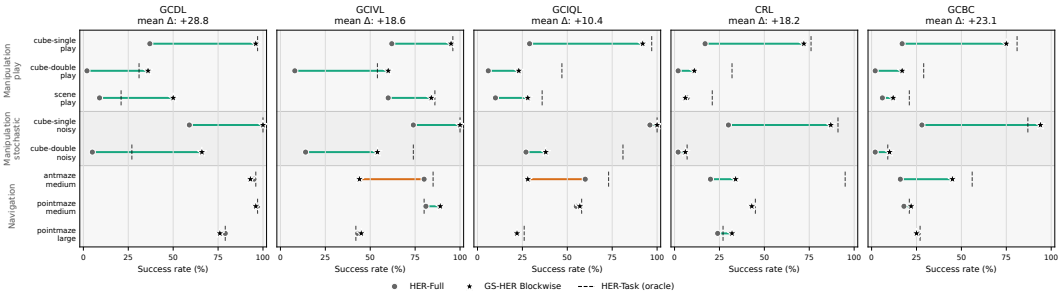


Figure 2: **Complete achieved outcomes can overconstrain downstream tasks.** Official OGBench success across five goal-conditioned learners. HER-Task provides an oracle diagnostic of the cost of full-state goal semantics, while GS-HER attempts to recover task-selective semantics without fixing one projection before training. Across manipulation, HER-Task improves over HER-Full in all $25/25$ backbone–task pairs and GS-HER Blockwise improves in $24/25$.

**Training and evaluation.** All methods are implemented in PyTorch following the corresponding OGBench reference implementations. For each environment, backbone, and goal formulation, we train five independent random seeds. Performance on the benchmark tasks is measured using the official OGBench success predicate and averaged over the same fixed evaluation protocol. These comparisons characterize a common 200k-update budget rather than asymptotic performance under substantially longer optimization; full details are provided in Appendix G.

### 4.2 A Consistent Task-Projection Gap in Manipulation

We first use HER-Task as a diagnostic for goal overspecification. HER-Task differs from HER-Full only by applying the official task projection to the goal-conditioning representation; the sampled reference and learner-specific relabeling metadata remain unchanged. Consequently, a large HER-Task–HER-Full gap indicates that requiring the complete achieved outcome is costly for the actual downstream task.

As shown in Figure 2, the pattern is striking in manipulation. Across the five manipulation environments and five learning backbones, HER-Task outperforms HER-Full in all $25/25$ backbone–task pairs. This observation is independent of GS-HER: replacing full-state goal conditioning with the oracle task projection consistently improves performance across imitation-, contrastive-, value-, and distance-based learners.

GS-HER recovers most of this benefit without fixing one task projection before training. Its annotation-free Blockwise variant improves over HER-Full in $24/25$ manipulation backbone–task pairs. Averaged across all manipulation pairs, official success increases from $24.4\%$ with HER-Full to $56.5\%$ with GS-HER, approaching the $60.0\%$ reported for the oracle HER-Task projection.

GS-HER exceeds HER-Task in several individual backbone–environment pairs; accordingly, HER-Task serves as a diagnostic projection rather than a performance ceiling.

Crucially, the same pattern does not appear when full outcome information is already aligned with task semantics. Across PointMaze, average success is $48.8\%$, $50.7\%$, and $49.2\%$ for HER-Full, GS-HER, and HER-Task, respectively. The negligible oracle-projection gain indicates little overspecification, and GS-HER correspondingly provides little benefit. This control argues against the weaker explanation that masking variables or using lower-dimensional goals is intrinsically beneficial. The advantage appears when the properties represented by the achieved outcome exceed those required by the downstream task.

AntMaze provides an important complementary case. HER-Task raises average success from $54.2\%$ to $81.0\%$, yet Blockwise reaches only $48.8\%$; Semantic partially recovers the gap at $68.8\%$ (Appendix A). Unlike PointMaze, AntMaze constrains only the 2-D root position within a much larger proprioceptive state with heterogeneous temporal scales, which may poorly match generic Blockwise queries. Thus, a large task-projection gap need not be recovered by a generic query distribution.

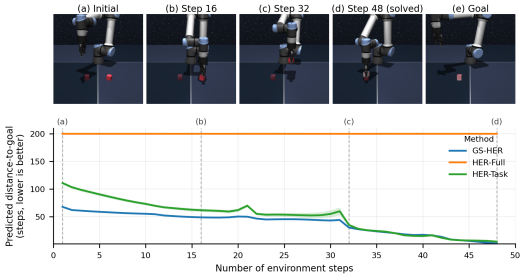


Figure 3: **Full-state goal distance can remain large after downstream success.** Along a successful `cube-single-noisy` rollout, the HER-Full GCDL estimate remains far from the complete reference outcome even as the official cube-position task is solved. HER-Task and GS-HER with the task query instead remove distinctions outside the downstream goal specification. Additional rollouts are provided in Appendix B.

### 4.3 Full-State Goal Distance Can Remain Large After Downstream Success

Performance alone does not reveal how the learned goal representation relates to downstream task progress. We therefore inspect GCDL, whose normalized value provides a distance-like temporal diagnostic under the conditioned goal representation.

Figure 3 shows one representative `cube-single-noisy` rollout. Under HER-Full, the predicted distance to the complete reference outcome remains near the maximum horizon even as the cube approaches and eventually satisfies the official position criterion. This need not indicate an inaccurate estimate: the full reference outcome still distinguishes robot configuration, gripper state, velocity, cube orientation, and other properties that the downstream cube-position task does not require.

HER-Task removes these distinctions through the fixed oracle projection. GS-HER with the cube-position query removes the same inactive distinctions while retaining one model across queries. Along this rollout, its distance-like estimate therefore follows downstream task progress more closely, but is not an exact first-hitting-time estimate for $\mathcal{G}(g, q)$.

The diagnostic illustrates goal overspecification at the representation level: a model can remain far from a complete reference outcome even after entering a set of outcomes that are equivalent for the downstream task. Additional rollouts across domains are provided in Appendix B.

### 4.4 One Goal-Conditioned Model, Multiple Evaluation Predicates

The oracle HER-Task results show that task-specific projections can avoid goal overspecification when the relevant success semantics are known in advance. General-purpose pretraining, however, should not require committing to a single projection: a property unconstrained by one task may define success in another. We therefore ask whether a single trained model can retain the same outcome information while changing which properties are required at inference time.

For each independently trained seed, we evaluate the same GS-HER Blockwise GCIQL checkpoint on `cube-single-noisy-v0` under seven goal predicates: cube position, cube yaw, cube pose, airborne cube position, end-effector position, end-effector pose, and end-effector pose with gripper opening. Within a seed, the model parameters remain fixed across all seven evaluations; only the inference-time query $q$ changes. All non-official reference outcomes are sampled from a held-out validation split, and every method is evaluated on the same fixed paired goal suite. Reported means and standard deviations aggregate the corresponding fixed-model evaluation across the five independently trained seeds. This experiment tests inference-time queryability within one environment and backbone. The evaluated query structures belong to the query family represented during GS-HER training, while the reference outcome values are held out. It therefore tests whether a fixed checkpoint can switch among represented query structures and generalize to new reference values.

The task-specific geometric thresholds used here are *evaluation criteria only*. They do not define GS-HER training rewards or query-dependent terminal labels. Training retains GCIQL's original target construction, including its future-trajectory demonstration witness.

Table 1: **One checkpoint, multiple evaluation predicates.** Success rate (%) with GCIQL on `cube-single-noisy`. GS-HER uses the same checkpoint for all predicates; HER-Full retains full-state goal semantics.

| PREDICATE | HER-FULL | GS-HER BLOCKWISE |
|---|---|---|
| Cube position (official) | $93\pm1$ | $\mathbf{100\pm0}$ |
| Cube yaw | $58\pm3$ | $57\pm11$ |
| Cube pose | $\mathbf{52\pm5}$ | $29\pm4$ |
| Airborne cube position | $65\pm4$ | $\mathbf{99\pm1}$ |
| End-effector position | $78\pm4$ | $\mathbf{98\pm1}$ |
| End-effector pose | $72\pm4$ | $\mathbf{99\pm1}$ |
| EE pose + gripper | $59\pm4$ | $\mathbf{85\pm2}$ |

Importantly, the evaluated query structures belong to the query family represented during GS-HER training; only the reference outcome values are held out. Accordingly, this experiment tests inference-time task specification and generalization to new reference values, rather than zero-shot generalization to arbitrary unseen query structures or compositions.

On the official cube-position task, all methods are already strong: HER-Full, HER-Task, and GS-HER achieve $93\%$, $100\%$, and $100\%$ success, respectively. The additional predicates therefore probe a different property: whether the same trained checkpoint can support different definitions of success while retaining the underlying outcome information.

As shown in Table 1, changing only the GS-HER query yields $99\%$ success on airborne cube position, $98\%$ on end-effector position, $99\%$ on end-effector pose, and $85\%$ on end-effector pose with gripper opening. These predicates constrain substantially different parts of the state, yet are executed by the same checkpoint. HER-Full likewise uses a single checkpoint, but its goal semantics always constrain the complete reference outcome. HER-Task can express a desired projection, but that projection is fixed during training and changing it requires a separately trained model.

Performance varies across predicates. Cube yaw is essentially unchanged relative to HER-Full ($57\%$ vs $58\%$ ), while cube pose is harder for GS-HER ($29\%$ versus $52\%$ for HER-Full). The central result is therefore queryability rather than uniform improvement across predicates: within the query family represented during training, the same checkpoint can impose different object- and robot-centric success constraints over the retained outcome information.

## 5 Discussion and Limitations

Our experiments support a simple distinction that is easy to overlook in goal-conditioned learning: the information used to describe an outcome and the information required to define task success need not coincide. Full-state hindsight is attractive precisely because it avoids deciding during pretraining which properties of an achieved outcome will matter later. However, when every retained property is also interpreted as something that must be reproduced, this generality can become a source of *goal overspecification*. Across all 25 manipulation backbone–task pairs, replacing full-state semantics with the oracle task projection improves downstream performance, and GS-HER recovers much of this gap without fixing that projection before training. Conversely, when outcome and task semantics are already closely aligned, as in PointMaze, the advantage largely disappears.

The distinction concerns goal constraints, not state availability. A property may be irrelevant to terminal success yet necessary for control, or may define another task entirely. GS-HER therefore preserves the complete current state while making only goal distinctions query-dependent. For future-trajectory goals, the value-based backbones retain the sampled offset as a one-sided reachability witness, $g = s_{t+k} \Rightarrow \tau_{\mathcal{G}(g,q)}(s_t) \leq k$. We do not recompute query-specific first hits, so this certificate is sound but potentially non-tight; doing so is an orthogonal extension to the representation-level intervention studied here.

**Beyond factored states.** We study factored state spaces because they make retained outcome information and task-required constraints explicit and experimentally controllable. Extending this

principle to pixels, learned latent representations, relational representations, or language requires learning or grounding the relevant task-dependent factors through object structure, latent variables, relations, or language-conditioned predicates. Factored-state GS-HER provides a controlled setting in which to isolate this information–constraint distinction.

**Scope of queryability.** The behaviors available at inference remain determined by offline data coverage and by the family of task specifications represented during training. Our multi-predicate evaluation therefore establishes inference-time queryability within that represented family. The AntMaze results further show that query structure matters. Its task constrains only root position within a much larger proprioceptive state whose factors evolve at different temporal scales; the stronger Semantic result is consistent with Blockwise poorly matching this structure.

**Implications for general-purpose robot learning.** The broader lesson is that outcome completeness and task specificity should be treated as separate design choices. General-purpose pretraining benefits from retaining rich descriptions of experience precisely because downstream tasks are not known in advance. But retaining a property should not imply requiring every future task to reproduce it. Put differently, a general-purpose learner should retain more information about an outcome than any individual task should require it to reproduce. The distinction between what an outcome describes and what a task requires is therefore not merely an implementation detail of hindsight relabeling, but a broader design principle for reusable goal-conditioned learning: representation should preserve future flexibility, while task semantics should determine which parts of that representation become behavioral requirements.

## 6 Conclusion

We studied *goal overspecification*, the mismatch that arises when a complete achieved outcome serves simultaneously as a rich description of experience and as the exact specification of task success. Across five heterogeneous goal-conditioned learners, the oracle task projection yields higher mean manipulation success than full-state semantics in all $25/25$ backbone–task pairs, while the gap becomes negligible on PointMaze when outcome and task semantics are already aligned.

GS-HER separates these roles through a query-conditioned goal interface. A query defines which outcome distinctions are relevant, while the representation is invariant to the remaining coordinates; the base learner's hindsight sampling, supervision, and learning objective remain unchanged.

This representation-level intervention recovers a substantial fraction of the oracle projection gap, supporting a broader design principle for reusable goal-conditioned learning: retain outcome information broadly, but identify goals only up to the distinctions that the current task requires.

### AI use statement

In this work, we used generative AI tools exclusively to assist with the writing and presentation of the manuscript, including language polishing, clarity, organization, and stylistic improvements to author-provided content. Generative AI tools were not used to generate experimental data or results, implement or execute experiments, or determine the scientific conclusions of the work. All AI-assisted text was reviewed, edited, and verified by the authors for technical accuracy, appropriate attribution, and consistency with the underlying experiments and results. The authors take full responsibility for the final content of the paper, including all text and claims produced with the aid of generative AI.

### Reproducibility statement

We document the method, experimental protocol, query distributions, architectures, training setup, evaluation procedure, and principal implementation choices in the main paper and appendix. The appendix additionally reports the complete per-environment results, multi-predicate evaluation protocol, and model-capacity details. All experiments use publicly available OGBench environments and datasets. Source code, complete configurations, task projections, and training and evaluation scripts will be released upon de-anonymization to enable exact reproduction of the reported experiments.

### ACKNOWLEDGMENTS

Funding will be detailed upon de-anonymization.

## REFERENCES

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, OpenAI Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. *Advances in neural information processing systems*, 30, 2017.

Manfred Eppe, Sven Magg, and Stefan Wermter. Curriculum goal masking for continuous deep reinforcement learning. *arXiv preprint arXiv:1809.06146*, 2018.

Ben Eysenbach, Xinyang Geng, Sergey Levine, and Russ R Salakhutdinov. Rewriting history with inverse rl: Hindsight inference for policy improvement. *Advances in neural information processing systems*, 33:14783–14795, 2020.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Russ R Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. *Advances in Neural Information Processing Systems*, 35:35603–35620, 2022.

Justin Fu, Anoop Korattikara, Sergey Levine, and Sergio Guadarrama. From language to goals: Inverse reinforcement learning for vision-based instruction following. *arXiv preprint arXiv:1902.07742*, 2019.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning, 2020. URL `https://arxiv.org/abs/1912.06088`.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. *arXiv preprint arXiv:2110.06169*, 2021.

John Banister Lanier. Curiosity-driven multi-criteria hindsight experience replay. Master's thesis, University of California, Irvine, 2019.

Alexander Li, Lerrel Pinto, and Pieter Abbeel. Generalized hindsight for reinforcement learning. *Advances in neural information processing systems*, 33:7754–7767, 2020.

Corey Lynch, Mohi Khansari, Ted Xiao, Vikash Kumar, Jonathan Tompson, Sergey Levine, and Pierre Sermanet. Learning latent plans from play. In *Conference on robot learning*, pp. 1113–1132. Pmlr, 2020.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. Liv: Language-image representations and rewards for robotic control. In *International Conference on Machine Learning*, pp. 23301–23320. PMLR, 2023.

Vivek Myers, Andre Wang He, Kuan Fang, Homer Rich Walke, Philippe Hansen-Estruch, Ching-An Cheng, Mihai Jalobeanu, Andrey Kolobov, Anca Dragan, and Sergey Levine. Goal representations for instruction following: A semi-supervised language interface to control. In *Conference on Robot Learning*, pp. 3894–3908. PMLR, 2023.

Ashvin V Nair, Vitchyr Pong, Murtaza Dalal, Shikhar Bahl, Steven Lin, and Sergey Levine. Visual reinforcement learning with imagined goals. *Advances in neural information processing systems*, 31, 2018.

Suraj Nair, Eric Mitchell, Kevin Chen, Silvio Savarese, Chelsea Finn, et al. Learning language-conditioned robot behavior from offline data and crowd-sourced annotation. In *Conference on Robot Learning*, pp. 1303–1315. PMLR, 2022.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. Hiql: Offline goal-conditioned rl with latent states as actions. *Advances in Neural Information Processing Systems*, 36:34866–34891, 2023.

Seohong Park, Kevin Frans, Benjamin Eysenbach, and Sergey Levine. Ogbench: Benchmarking offline goal-conditioned rl. In *International Conference on Learning Representations*, volume 2025, pp. 94937–94982, 2025.

Seohong Park, Deepinder Mann, and Sergey Levine. Dual goal representations. In *International Conference on Learning Representations*, volume 2026, pp. 153307–153324, 2026.

Vitchyr H Pong, Murtaza Dalal, Steven Lin, Ashvin Nair, Shikhar Bahl, and Sergey Levine. Skew-fit: State-covering self-supervised reinforcement learning. *arXiv preprint arXiv:1903.03698*, 2019.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International conference on machine learning*, pp. 1312–1320. PMLR, 2015.

Simon Ståhlberg and Hector Geffner. First-order representation languages for goal-conditioned rl. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 40, pp. 36394–36402, 2026.

# A Full OGBench Results

Table 2 reports the complete OGBench results underlying Fig. 2. For each backbone and environment, we compare full-state hindsight (HER-Full), the oracle task-specific projection (HER-Task), and two GS-HER query distributions. All values denote official binary success rate (%), averaged over five independently trained seeds.

The complete table makes two patterns from the main text explicit. First, across the five manipulation environments and five learning backbones, HER-Task outperforms HER-Full in all $25/25$ backbone–task pairs. Thus, the cost associated with complete goal semantics is not specific to a single learning objective. Second, GS-HER Blockwise improves over HER-Full in $24/25$ of these pairs without fixing the official task projection before training. In contrast, the differences are substantially smaller on PointMaze, where represented state and downstream goal semantics are already closely aligned.

# B Additional Goal-Overspecification Diagnostics

The qualitative diagnostic in the main paper uses GCDL to visualize how different goal semantics alter the learned notion of distance to a goal. Because the GCDL value estimate can be rescaled into an estimate of the remaining steps to the conditioned goal, it provides an interpretable view of the objectives induced by HER-Full, HER-Task, and GS-HER.

Figure 4 presents six additional evaluation rollouts, with two examples each from `cube-single-noisy`, `scene-play`, and `antmaze-medium`. These examples extend the single representative rollout shown in the main text and illustrate the diagnostic across domains with different relationships between represented outcome information and downstream task semantics.

In the manipulation domains, the plots highlight that progress under the downstream task predicate can differ substantially from progress toward the complete reference state. HER-Task uses the oracle task-projected conditioning representation, whereas GS-HER evaluated with the official task query removes the same inactive distinctions through its query-conditioned interface. Consequently, a large HER-Full distance need not indicate an incorrect value estimate: the agent may be close to satisfying, or may already satisfy, the downstream task while still being far from reproducing the complete reference outcome.

The navigation examples illustrate that this mismatch is domain-dependent, rather than an inherent advantage of using fewer goal coordinates. When represented outcome information and downstream task semantics are more closely aligned, the discrepancy between full-state and task-selective distance estimates becomes correspondingly smaller.

Table 2: Full OGBench results. Official binary success rate (%), reported as mean ± standard deviation over five training seeds. HER-Full uses complete achieved states as goals; HER-Task uses the oracle projection corresponding to the official task predicate; GS-HER Blockwise uses annotation-free queries; and GS-HER Semantic uses privileged factor-level queries. HER-Task improves over HER-Full in all $25/25$ manipulation backbone–task pairs, while GS-HER Blockwise improves in $24/25$.

| BACKBONE | TASK | HER Full | HER Task | GS-HER Blockwise | GS-HER Semantic |
|---|---|---|---|---|---|
| GCDL | cube-single-play-v0 | 37 ± 4 | **97 ± 2** | **96 ± 1** | **96 ± 1** |
| | cube-single-noisy-v0 | 59 ± 5 | **100 ± 0** | **100 ± 0** | **100 ± 0** |
| | cube-double-play-v0 | 2 ± 1 | 31 ± 5 | 36 ± 11 | **43 ± 4** |
| | cube-double-noisy-v0 | 5 ± 1 | 27 ± 8 | **66 ± 7** | **70 ± 8** |
| | scene-play-v0 | 9 ± 2 | 21 ± 4 | **50 ± 6** | **50 ± 6** |
| | pointmaze-medium-navigate-v0 | **97 ± 2** | **97 ± 1** | **96 ± 5** | **98 ± 2** |
| | pointmaze-large-navigate-v0 | **79 ± 7** | **79 ± 8** | **76 ± 9** | **76 ± 7** |
| | antmaze-medium-navigate-v0 | **95 ± 2** | **96 ± 1** | **93 ± 2** | **96 ± 1** |
| GCIVL | cube-single-play-v0 | 62 ± 6 | **96 ± 3** | **95 ± 2** | **96 ± 0** |
| | cube-single-noisy-v0 | 74 ± 6 | **100 ± 0** | **100 ± 0** | **100 ± 0** |
| | cube-double-play-v0 | 8 ± 2 | 54 ± 10 | **60 ± 7** | **60 ± 4** |
| | cube-double-noisy-v0 | 14 ± 2 | **74 ± 7** | 54 ± 9 | 50 ± 3 |
| | scene-play-v0 | 60 ± 9 | **86 ± 2** | **84 ± 3** | **84 ± 3** |
| | pointmaze-medium-navigate-v0 | 81 ± 7 | 80 ± 8 | 89 ± 9 | **97 ± 4** |
| | pointmaze-large-navigate-v0 | **43 ± 5** | 42 ± 5 | **45 ± 4** | 39 ± 1 |
| | antmaze-medium-navigate-v0 | 80 ± 4 | **85 ± 3** | 44 ± 13 | 65 ± 4 |
| GCIQL | cube-single-play-v0 | 29 ± 2 | **97 ± 3** | **92 ± 3** | 91 ± 6 |
| | cube-single-noisy-v0 | **96 ± 2** | **100 ± 0** | **100 ± 0** | **99 ± 1** |
| | cube-double-play-v0 | 6 ± 3 | **47 ± 4** | 23 ± 4 | 15 ± 3 |
| | cube-double-noisy-v0 | 27 ± 4 | **81 ± 6** | 38 ± 7 | 28 ± 3 |
| | scene-play-v0 | 10 ± 0 | **36 ± 3** | 28 ± 2 | 28 ± 2 |
| | pointmaze-medium-navigate-v0 | 55 ± 7 | **58 ± 3** | **57 ± 6** | **60 ± 0** |
| | pointmaze-large-navigate-v0 | 22 ± 3 | **26 ± 7** | 22 ± 7 | 21 ± 12 |
| | antmaze-medium-navigate-v0 | 60 ± 6 | **73 ± 5** | 28 ± 3 | 35 ± 5 |
| CRL | cube-single-play-v0 | 17 ± 2 | **76 ± 3** | **72 ± 4** | 69 ± 5 |
| | cube-single-noisy-v0 | 30 ± 3 | **91 ± 4** | **87 ± 7** | 80 ± 3 |
| | cube-double-play-v0 | 2 ± 1 | **32 ± 5** | 11 ± 2 | 12 ± 3 |
| | cube-double-noisy-v0 | 2 ± 1 | **7 ± 2** | 6 ± 4 | 6 ± 4 |
| | scene-play-v0 | 7 ± 2 | **21 ± 5** | 6 ± 1 | 6 ± 1 |
| | pointmaze-medium-navigate-v0 | **43 ± 11** | 35 ± 8 | **43 ± 10** | **45 ± 12** |
| | pointmaze-large-navigate-v0 | 24 ± 4 | 27 ± 12 | **32 ± 5** | **22 ± 6** |
| | antmaze-medium-navigate-v0 | 20 ± 4 | **95 ± 1** | 34 ± 10 | 71 ± 4 |
| GCBC | cube-single-play-v0 | 17 ± 4 | **81 ± 4** | 75 ± 5 | 73 ± 3 |
| | cube-single-noisy-v0 | 28 ± 4 | 87 ± 4 | **94 ± 2** | 88 ± 7 |
| | cube-double-play-v0 | 2 ± 1 | **29 ± 4** | 17 ± 5 | 13 ± 4 |
| | cube-double-noisy-v0 | 2 ± 1 | 9 ± 3 | **10 ± 4** | 4 ± 2 |
| | scene-play-v0 | 6 ± 1 | **21 ± 4** | 12 ± 2 | 11 ± 2 |
| | pointmaze-medium-navigate-v0 | 18 ± 3 | 21 ± 5 | **22 ± 6** | **23 ± 6** |
| | pointmaze-large-navigate-v0 | 26 ± 8 | 27 ± 6 | 25 ± 5 | **30 ± 7** |
| | antmaze-medium-navigate-v0 | 16 ± 2 | 56 ± 2 | 45 ± 7 | **77 ± 3** |

### B.1 Aggregate Relation to the Oracle Projection Gap

We next examine whether the benefit of GS-HER is larger in environments where full-state goal semantics are more strongly misaligned with the downstream task. As an empirical diagnostic, we use the performance gap between HER-Full and HER-Task. HER-Task replaces the complete goal-conditioning representation with the fixed oracle task projection while preserving the underlying learner and relabeling supervision. Because this also changes conditioning dimensionality, we interpret the HER-Task gain as a projection-gap diagnostic rather than an isolated causal estimate of goal overspecification.

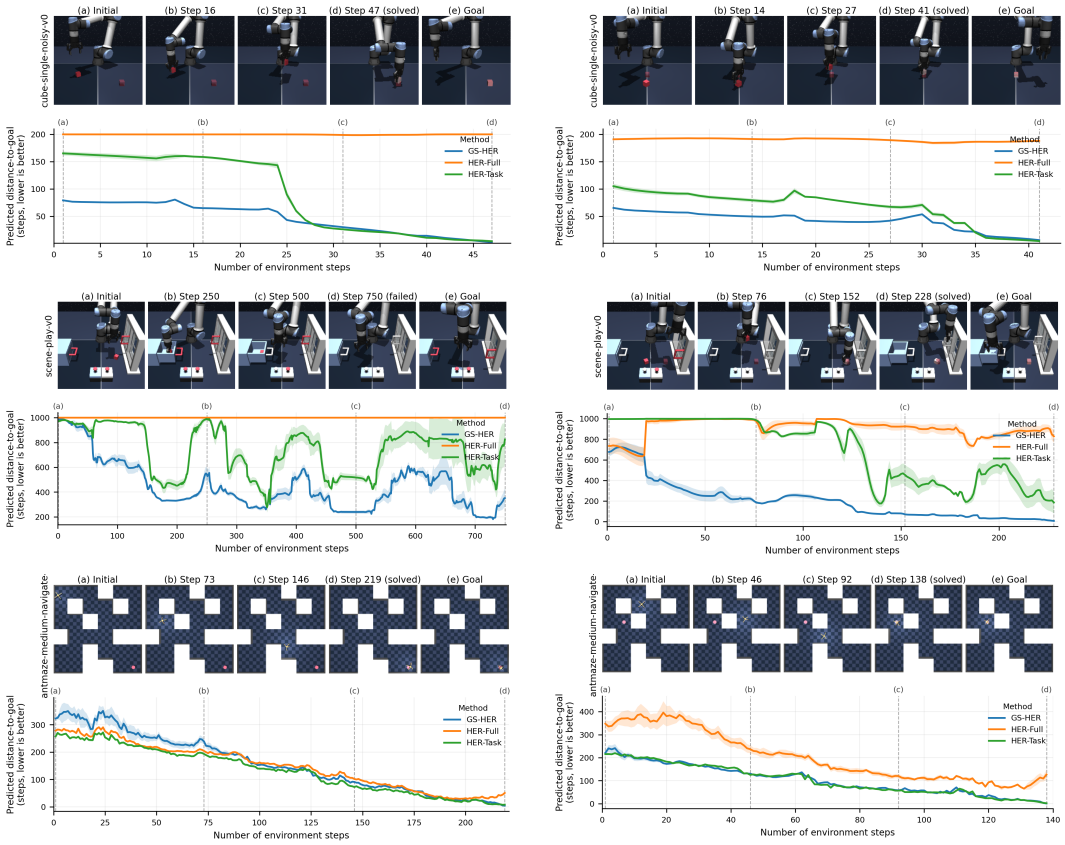


Figure 4: **Additional qualitative distance-to-goal diagnostics.** Two evaluation rollouts are shown for each of `cube-single-noisy` (top), `scene-play` (middle), and `antmaze-medium` (bottom). GCDL predictions are rescaled into estimated remaining steps to the conditioned goal. HER-Full is conditioned on the complete reference state, whereas HER-Task and GS-HER with the official task query remove inactive goal distinctions. The resulting GCDL values are distance-like diagnostics rather than exact first-hitting-time estimates. The divergence between these distance estimates varies with the alignment between represented outcome information and the downstream task predicate.

Figure 5 compares this oracle-projection gain with the gain obtained by GS-HER. Each point corresponds to one OGBench environment, with performance averaged across the base goal-conditioned learners. The two quantities exhibit a strong positive association: environments in which the oracle task projection yields a large improvement over full-state hindsight also tend to be those in which GS-HER provides the largest gains. Conversely, when replacing full-state semantics with the oracle task projection provides little benefit, GS-HER also tends to yield only small improvements.

This pattern is consistent with the central hypothesis of the paper: task-selective goal semantics are most useful when a complete outcome description imposes requirements beyond those defining downstream success. Because both quantities are measured relative to the same HER-Full baseline, however, the observed correlation should be interpreted as a descriptive aggregate diagnostic rather than as an independent causal test of goal overspecification. The analysis therefore complements, rather than replaces, the direct per-task comparisons and rollout-level diagnostics reported elsewhere.

AntMaze is the principal exception to this aggregate trend: it exhibits a large oracle-projection gain but a negative Blockwise gain. Its 2-D navigation goal is embedded in a much larger proprioceptive state with heterogeneous temporal scales, potentially mismatching generic Blockwise queries. The stronger Semantic result supports the importance of query structure.

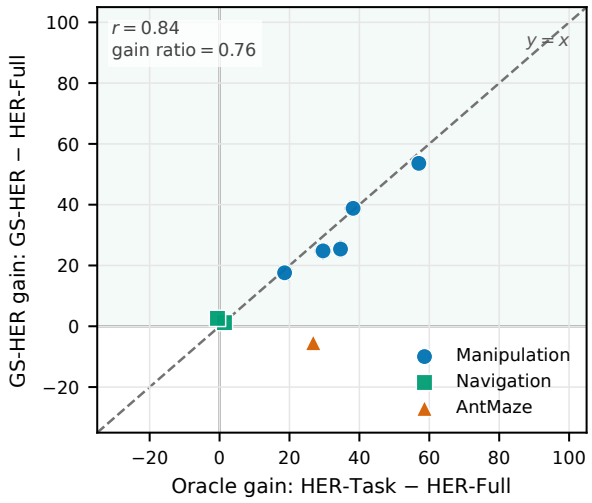


Figure 5: **Aggregate association between the oracle-projection gap and GS-HER gains.** Each point represents one OGBench environment, averaged across learners. The $x$-axis shows HER-Task–HER-Full and the $y$-axis GS-HER–HER-Full. The positive association is consistent with GS-HER helping more where the task-projection gap is larger, with AntMaze a clear exception. Because both axes subtract the same HER-Full baseline, the correlation is descriptive rather than an independent causal test.

## C Multi-Predicate Evaluation Protocol

The success functions defined in this appendix are used only to score the non-official evaluation tasks. They are not part of GS-HER training and are not used to construct rewards, success labels, or value targets. GS-HER training uses the unchanged learner-specific supervision described in Appendix G.3.

This appendix specifies the evaluation protocol underlying the multi-predicate experiment in Sec. 4.4 and Table 1. The experiment evaluates whether a fixed GS-HER checkpoint can expose different success semantics by changing only the inference-time query $q$.

We use GCIQL on `cube-single-noisy-v0`, where performance on the official cube-position predicate is already near saturation. This setting therefore isolates queryability from raw competence on the official benchmark task. For GS-HER, model parameters remain fixed across all predicates. HER-Full is likewise evaluated using a single checkpoint, but retains full-state goal semantics.

All non-official reference outcomes are drawn from a held-out validation split. Reference values are therefore unseen during training, while the evaluated query structures belong to the query family represented during GS-HER training. The experiment tests inference-time task specification rather than generalization to arbitrary unseen query structures.

### C.1 Environment and State Factorization

We use the state-based `cube-single-noisy-v0` environment. The observation has dimension $D = 28$ and contains robot proprioception, end-effector state, gripper state, and cube state. Table 3 reports the coordinate groups used to construct the evaluation predicates.

All predicate tests are computed in the observation coordinates returned by the environment, before applying the normalizer used by the learned policy. This distinction matters because OGBench internally scales Cartesian coordinates. Let $x_{\text{raw}}$ denote a Cartesian coordinate in meters and $x_{\text{obs}}$ its corresponding observation coordinate. For object and end-effector positions,

$$x_{\text{obs}} = 10\,(x_{\text{raw}} - x_{\text{center}})\,. \tag{5}$$

Table 3: State layout for `cube-single-noisy-v0`

| GROUP | DESCRIPTION | INDICES |
|---|---|---|
| `joint_pos` | Robot joint positions | 0:5 |
| `joint_vel` | Robot joint velocities | 6:11 |
| `eff_pos` | End-effector Cartesian position | 12:14 |
| `eff_yaw` | End-effector yaw as $(\cos, \sin)$ | 15:16 |
| `gripper_opening` | Gripper opening | 17 |
| `gripper_contact` | Gripper contact signal | 18 |
| `block0_pos` | Cube Cartesian position | 19:21 |
| `block0_quat` | Cube quaternion | 22:25 |
| `block0_yaw` | Cube yaw as $(\cos, \sin)$ | 26:27 |

The official cube-position criterion uses a Euclidean radius of $0.04\,\mathrm{m}$ in Cartesian coordinates, corresponding to

$$\epsilon_{\mathrm{pos}} = 10 \times 0.04 = 0.40 \tag{6}$$

in observation space.

Yaw is represented as $(\cos\theta, \sin\theta)$. We therefore measure angular error using chord distance,

$$\begin{aligned} d_{\mathrm{yaw}}(\theta, \theta^\star) &= \left\| \begin{bmatrix} \cos\theta \\ \sin\theta \end{bmatrix} - \begin{bmatrix} \cos\theta^\star \\ \sin\theta^\star \end{bmatrix} \right\|_2 \\ &= 2\sin\left(\frac{|\Delta\theta|}{2}\right), \end{aligned} \tag{7}$$

where $\Delta\theta$ denotes the wrapped angular difference. A $20^\circ$ angular tolerance therefore corresponds to

$$\epsilon_{\mathrm{yaw}} = 2\sin(10^\circ) \approx 0.347. \tag{8}$$

### C.2 Evaluation Predicates

Each evaluation task is specified by a complete reference outcome $g \in \mathbb{R}^D$ and a binary query $q \in \{0,1\}^D$. The reference retains the full achieved state, while the query specifies which properties define success for the current task. Coordinates not selected by $q$ remain available in the state and reference representations but are not constrained by that task.

Because the state combines heterogeneous physical quantities, evaluation does not apply a single Euclidean threshold over all active coordinates. Instead, each active semantic group is tested using an appropriate metric. For a query containing $M_q$ terms, success is defined as

$$c_{g,q}(s) = \prod_{m=1}^{M_q} \mathbb{I}\left[d_m(s, g) \le \epsilon_m\right], \tag{9}$$

where $d_m$ is an $\ell_2$ distance for Cartesian quantities, an absolute distance for scalar quantities, or the chord distance in Eq. 7 for yaw.

Table 4 defines the seven predicates used in the experiment.

### C.3 Held-Out Reference Outcome Suite

All non-official reference outcomes are sampled from a held-out validation split that is never used to train any evaluated model. The evaluation suite is generated once and reused unchanged across methods, training checkpoints, and random seeds. This paired construction prevents differences in sampled reference difficulty from confounding comparisons between methods.

The suite contains

$$7 \text{ predicates} \times 5 \text{ official task IDs} \times 20 \text{ episodes per task} = 700 \tag{10}$$

Table 4: Goal predicates used in the multi-predicate evaluation. Thresholds are expressed in OGBench observation coordinates before policy normalization.

| PREDICATE | ACTIVE GROUPS | SUCCESS TEST | REFERENCE SOURCE |
|---|---|---|---|
| cube_pos | block0_pos | $\|s_{19:21} - g_{19:21}\|_2 \leq 0.40$ | Official environment goal |
| cube_yaw | block0_yaw | Yaw chord distance $\leq 0.347$ | Held-out tabletop states |
| cube_pose | block0_pos, block0_yaw | Position $\leq 0.40$ and yaw $\leq 0.347$ | Held-out tabletop states |
| airborne_cube_pos | block0_pos | $\|s_{19:21} - g_{19:21}\|_2 \leq 0.30$ | Held-out airborne states |
| ee_pos | eff_pos | $\|s_{12:14} - g_{12:14}\|_2 \leq 0.40$ | Held-out states |
| ee_pose | eff_pos, eff_yaw | Position $\leq 0.40$ and yaw $\leq 0.347$ | Held-out states |
| ee_pose_gripper | eff_pos, eff_yaw, gripper_opening | Position $\leq 0.40$, yaw $\leq 0.347$, and opening $\leq 0.10$ | Held-out states |

Table 5: Cube-height statistics of the fixed reference outcome suite

| PREDICATE | MIN. $g_{21}$ | MEAN $g_{21}$ | MAX. $g_{21}$ |
|---|---|---|---|
| cube_pos | 0.200 | 0.200 | 0.200 |
| cube_yaw | 0.190 | 0.212 | 0.322 |
| cube_pose | 0.180 | 0.210 | 0.316 |
| airborne_cube_pos | 0.649 | 1.325 | 2.045 |

episode specifications, giving $N = 100$ episodes per predicate.

For cube_pos, we use the official goal provided by the environment. For every other predicate, a complete reference state is sampled from the held-out validation bank. Candidate references that are already too close to the initial observation under the corresponding predicate are rejected using a normalized minimum distance margin of 2.0.

For tabletop object predicates, reference outcomes are restricted to states where the cube remains near table height:

$$0.10 \leq g_{21} \leq 0.35, \tag{11}$$

where coordinate 21 is the cube vertical position in observation space. Under Eq. 5, this interval corresponds to approximately 1–3.5 cm of vertical offset in the underlying Cartesian coordinates. This prevents cube_yaw and cube_pose from being inadvertently defined primarily by lifted-cube states.

For airborne_cube_pos, we instead require

$$g_{21} \geq 0.60, \tag{12}$$

corresponding to a vertical offset of at least 0.06 m. Table 5 reports the resulting cube-height statistics of the fixed suite. These statistics are determined before model evaluation and are independent of the evaluated method.

### C.4 Paired Evaluation Protocol and Metrics

Evaluation is paired across methods. For every episode specification, each method receives the same environment seed, official task ID, predicate, and complete reference outcome. The environment is reset using the specified task_id and env_seed, while its original dynamics, observation space, action space, and episode horizon remain unchanged.

The reward returned by the environment is ignored for this evaluation. Success is instead determined by the predicate in Eq. 9. For each predicate and method, we report

$$\text{Success} = 100 \cdot \frac{1}{N} \sum_{i=1}^{N} \mathbb{I}\left[\exists t \le T : c_{g_i,q_i}(s_t) = 1\right], \tag{13}$$

where $N = 100$ episodes per predicate.

Success is latched: an episode is counted as successful once the requested predicate is satisfied at any timestep, even if the trajectory subsequently leaves the corresponding goal set. The fixed suite is stored once and loaded unchanged for every method and training seed. Results reported in Table 1 are mean $\pm$ standard deviation over independently trained checkpoints.

# D Query Distributions and Goal Predicates

GS-HER samples a binary query $q \sim p_Q(q)$, $q \in \{0,1\}^D$, $\|q\|_0 \ge 1$, for each relabeled reference outcome $g$. The goal sampler itself is unchanged: $g$ is sampled identically across HER-Full, HER-Task, and GS-HER, while $q$ determines which retained properties of $g$ constitute constraints for the current goal. We evaluate two query distributions: an annotation-free Blockwise distribution that uses no explicit semantic factor labels, and an oracle Semantic distribution using known state factors.

## D.1 Blockwise Queries

Our default GS-HER variant requires only the ordering of the state vector. Queries are sampled from

$$p_Q^{\text{blk}} = 0.15\, p_{\text{full}} + 0.15\, p_{\text{rand}} + 0.70\, p_{\text{block}}. \tag{14}$$

For $p_{\text{full}}$, all coordinates are active, $q = \mathbf{1}$, retaining standard full-state reaching in the training distribution.

For $p_{\text{rand}}$, the number of active coordinates is sampled as

$$K \sim \text{Unif}\{1, \dots, D\}, \tag{15}$$

after which $K$ coordinates are selected uniformly without replacement. This provides unstructured coordinate subsets independently of state-vector locality.

For $p_{\text{block}}$, we sample $M \sim \text{Unif}\{1, \dots, 4\}$ contiguous spans. For each span,

$$\rho \sim \mathcal{U}(0.15, 0.40), \qquad L = \lfloor \rho D \rfloor, \tag{16}$$

with $L$ clipped to $[1, D]$. Its start index is sampled uniformly from $\{0, \dots, D - L\}$, and the final query is the union of the $M$ spans. Spans may overlap. This construction exposes the learner to structured subsets without requiring semantic annotations of the state coordinates.

Blockwise therefore assumes no semantic labels for individual coordinates, but it is not invariant to an arbitrary permutation of the state vector. Its contiguous-span component encodes a locality prior: coordinates that are nearby in the observation vector are more likely to be queried jointly. OGBench commonly stores coordinates belonging to the same physical factor in contiguous groups, so this prior can align with environment structure. We make this assumption explicit and do not treat Blockwise as fully structure-agnostic.

## D.2 Semantic Queries

GS-HER Semantic uses the environment-provided state factorization. Let $\{\mathcal{I}_1, \dots, \mathcal{I}_G\}$ denote the coordinate groups associated with the $G$ semantic factors. A query is constructed by first sampling the number $K$ of active groups.

With probability $0.6$, we use a sparse-biased log-space sampler:

$$u \sim \mathcal{U}(0, \log(G+1)), \qquad K = \lfloor e^u \rfloor, \tag{17}$$

clipped to $[1, G]$. With probability $0.4$,

$$K \sim \mathrm{Unif}\{1, \ldots, G\}. \tag{18}$$

We then select $K$ semantic groups uniformly without replacement and activate the union of their coordinates. For $G = 1$, the single group is always selected.

This mixture favors compact predicates while retaining coverage of multi-factor goals. Unlike Blockwise queries, it uses privileged knowledge of which coordinates form meaningful physical factors and is therefore treated as an oracle-query variant.

### D.3 Sampling and Inference

For efficiency, each training run precomputes a cache of $131{,}072$ queries for its configured strategy and samples uniformly from that cache during training.

At inference time, $q^\star$ is supplied explicitly and changes which properties of the reference outcome are exposed as goal constraints without changing the model weights. For example, different queries can request cube position, cube pose, or end-effector pose from the same reference outcome and checkpoint. The complete current state remains available to the controller; $q^\star$ modifies the goal specification rather than the state information used for control. Accordingly, our multi-predicate experiment tests inference-time queryability and generalization to held-out reference values.

## E Base Goal-Conditioned Learners

GS-HER is evaluated as a relabeling and goal-conditioning wrapper rather than as a new offline RL objective. To test whether the observed effect depends on a particular learning rule, we apply the same goal-semantics variants to five heterogeneous offline goal-conditioned learners. Below we summarize their role in our experiments; complete algorithmic derivations are available in the corresponding original work.

**Goal-conditioned behavioral cloning (GCBC).** GCBC is the simplest goal-conditioned learner. It trains a policy by supervised imitation, conditioning each state-action pair on a future state from the same trajectory as the goal (Lynch et al., 2020; Ghosh et al., 2020; Park et al., 2025). Since GCBC does not perform temporal-difference learning, it cannot explicitly propagate reachability beyond demonstrated transitions, but it provides a useful imitation-based baseline for testing whether query-conditioned goal representations can be learned from offline trajectories.

**Contrastive reinforcement learning (CRL).** CRL learns goal-conditioned reachability through a contrastive objective rather than a Bellman backup (Eysenbach et al., 2022; Park et al., 2025). It trains a score function to distinguish future goals from random goals, which can be interpreted as learning a Monte Carlo goal-conditioned value estimate. In OGBench, CRL is a representative contrastive GCRL method and is particularly useful for testing whether GS-HER is compatible with non-Bellman, representation-learning-based objectives.

**Goal-conditioned implicit value learning (GCIVL).** GCIVL is a value-only goal-conditioned variant of implicit Q-learning (Kostrikov et al., 2021; Park et al., 2023; 2025). It fits a goal-conditioned value function using expectile regression and extracts a policy with advantage-weighted regression. In OGBench, GCIVL is used as a representative value-based offline GCRL method that estimates goal reachability without learning an explicit action-value function. Future-trajectory goals additionally use the shared one-sided demonstration-witness bound described in Appendix G.3.

**Goal-conditioned implicit Q-learning (GCIQL).** GCIQL is the action-value counterpart of GCIVL. It learns both a goal-conditioned value function and a goal-conditioned Q-function using an IQL-style objective (Kostrikov et al., 2021; Park et al., 2023; 2025). The policy is then extracted using a DDPG+BC-style objective, following the OGBench implementation. GCIQL provides a strong value-based baseline, especially in manipulation environments. Its critic targets use the same trajectory-goal demonstration-witness bound described in Appendix G.3.

**Goal-conditioned distance learning (GCDL).** GCDL is a simple diagnostic distance-learning baseline introduced in this work. It learns a bounded value function whose output can be interpreted as a normalized negative distance-to-goal, and then extracts a policy using advantage-weighted regression. Unlike GCIVL, GCIQL, or CRL, GCDL is not intended as a new competitive offline GCRL algorithm. We include it because its value predictions can be directly rescaled into estimated remaining steps-to-goal, making it useful for visualizing how different relabeling schemes shape the learned value landscape.

### E.1 Goal-Conditioned Distance Learning

We include Goal-Conditioned Distance Learning (GCDL) as a simple diagnostic offline GCRL backbone. GCDL is not a contribution of this work; it is introduced because its value estimate provides an interpretable temporal reachability score calibrated by the sampled trajectory offsets. This makes it useful for visualizing how different relabeling schemes shape the learned value geometry. Importantly, GCDL uses the same unified query-conditioned goal interface as the other backbones in this paper.

#### E.1.1 Value Parameterization

GCDL uses a double goal-conditioned value estimator on a normalized temporal distance scale. A value near $0$ represents the terminal boundary induced by the training targets, while increasingly negative values correspond to larger predicted temporal distance under the inherited relabeling supervision. With maximum relabeling horizon $H_{\max}$, we interpret

$$\hat{d}_{\theta,j}(s,g,q) = -H_{\max} V_{\theta,j}(s,g,q).$$

Training targets are clipped to $[-1,0]$, corresponding to distances between zero and approximately $H_{\max}$ steps.

#### E.1.2 Temporal-Distance Value Learning

GCDL retains the reference-outcome metadata produced by the common relabeling sampler. Let $m_{\mathrm{curr}}$ indicate that the sampled reference is the current state, $m_{\mathrm{traj}}$ indicate that it is a future state from the same trajectory, and $k_{\mathrm{traj}}$ denote the realized temporal offset for trajectory goals. These quantities are sampled before the GS-HER query and are not recomputed from $q$.

Define the normalized one-step cost

$$\delta = -\frac{1}{H_{\max}},$$

and let

$$\bar{V}(s,g,q) = \min_{j\in\{1,2\}} V_{\bar{\theta},j}(s,g,q)$$

denote the pessimistic estimate from the target value network.

The implementation also treats predictions sufficiently close to zero as a bootstrapped goal boundary. Define

$$b_t = \mathbb{I}\left[\bar{V}(s_t,g,q) > \frac{\delta}{2}\right], \qquad b_{t+1} = \mathbb{I}\left[\bar{V}(s_{t+1},g,q) > \frac{\delta}{2}\right].$$

The effective boundary indicators are then

$$h_t = m_{\mathrm{curr}} \vee b_t,$$

and

$$h_{t+1} = (m_{\mathrm{traj}} \wedge k_{\mathrm{traj}} = 1) \vee b_{t+1}.$$

The one-step temporal-distance target is

$$y_t = \begin{cases} 0, & h_t = 1, \\ \delta, & h_t = 0 \text{ and } h_{t+1} = 1, \\ \delta + \bar{V}(s_{t+1},g,q), & \text{otherwise.} \end{cases}$$

For a future-trajectory reference $g = s_{t+k_{\text{traj}}}$, the demonstrated suffix certifies $\tau_{\mathcal{G}(g,q)}(s_t) \le k_{\text{traj}}$. The implementation therefore lower-bounds the target by the corresponding demonstrated temporal value,

$$y_t \leftarrow \max\left(y_t, -\frac{k_{\text{traj}}}{H_{\max}}\right) \qquad \text{when } m_{\text{traj}} = 1.$$

Finally,

$$y_t \leftarrow \text{clip}(y_t, -1, 0),$$

and the two value estimators are trained by regression to the same target,

$$\mathcal{L}_V(\theta) = \mathbb{E}\left[\sum_{j=1}^{2} \left(V_{\theta,j}(s_t, g, q) - y_t\right)^2\right].$$

Importantly, no geometric query-conditioned success predicate is evaluated in this target construction. The query affects GCDL through the same goal-conditioning representation used by the other backbones, while the temporal supervision remains inherited from the common relabeling sampler. Consequently, the rescaled GCDL predictions used in our figures are distance-like diagnostics rather than exact first-hitting-time estimates for the full equivalence class $\mathcal{G}(g, q)$.

#### E.1.3 Policy Learning with Advantage-Weighted Regression

GCDL trains a query-conditioned policy using advantage-weighted regression as GCIVL from (Park et al., 2025). We compute the advantage in distance space and using the pessimistic value estimate.

#### E.1.4 Role in the Paper

GCDL is intended as an interpretable diagnostic backbone rather than as a new competitive offline RL algorithm. Its normalized value scale can be converted into an estimated remaining temporal distance, making it useful for visualizing how different goal-conditioning representations shape learned reachability estimates.

HER-Full conditions this estimator on complete reference outcomes. HER-Task uses the fixed oracle projection, while GS-HER uses the same temporal supervision with query-conditioned goal representations. Differences among their predicted distances therefore provide a diagnostic of how retaining or removing goal distinctions changes the learned geometry, rather than of different task-specific reward functions.

## F Evaluation Tasks Visualization

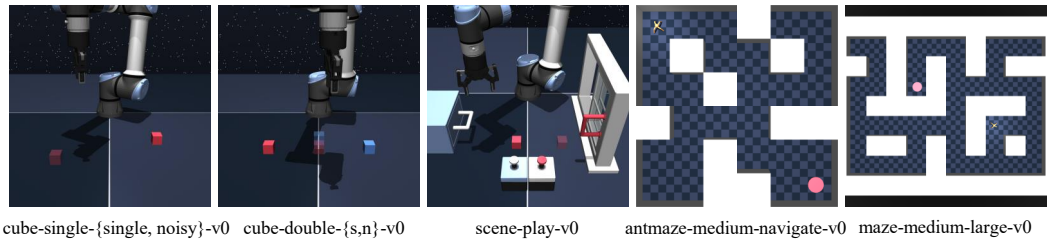


Figure 6: **Evaluation tasks**. State-based OGBench manipulation and navigation domains.

## G Implementation and Reproducibility Details

This appendix provides the architectural, optimization, sampling, and evaluation details used throughout our experiments. All five offline goal-conditioned learners are implemented in PyTorch using the official OGBench JAX implementations as the main algorithmic reference. Unless explicitly stated otherwise, we retain the backbone-specific OGBench hyperparameters and do not perform

method-specific hyperparameter sweeps. Across HER-Full, HER-Task, and GS-HER, we keep the architecture class, network depth, hidden widths, data, optimization protocol, and number of gradient updates fixed. The variants differ in the construction of the goal-conditioning input, with a small parameter-count difference induced by the additional query-conditioned input in GS-HER.

### G.1 Shared Goal-Conditioned Architecture

All policies, value functions, and critics use a common goal-conditioned multilayer perceptron, denoted by

$$\text{GC-MLP}_\theta(x, g_{\text{in}}),$$

where $x$ denotes the state input, or the state–action input for critics, and $g_{\text{in}}$ denotes the goal-conditioning input.

The conditioning input is first embedded by a shallow MLP followed by layer normalization,

$$h_g = \text{LayerNorm}\left(f_\theta^g(g_{\text{in}})\right),$$

with embedding dimension

$$d_{\text{emb}} = \max\left(8, \left\lfloor \frac{d_{g_{\text{in}}}}{2} \right\rfloor\right).$$

The resulting goal embedding is concatenated with the state or state–action input and processed by a second MLP,

$$y = f_\theta^{\text{out}}\left([x, h_g]\right).$$

All MLPs use GELU activations and hidden width $256$. Initialization, normalization, architectural depth, and hidden widths are kept fixed across relabeling variants for a given backbone.

**Goal-conditioning inputs.** The architectural template is shared across variants; only the construction of $g_{\text{in}}$ changes. For HER-Full,

$$g_{\text{in}}^{\text{Full}}(s, g) = [s, g].$$

For HER-Task, the same interface is applied after the fixed oracle task projection $\phi$,

$$g_{\text{in}}^{\text{Task}}(s, g) = [\phi(s), \phi(g)].$$

For GS-HER, given a query $q \in \{0, 1\}^D$, we construct

$$\bar{s}_q = q \odot s + (1 - q) \odot e_s, \qquad \bar{g}_q = q \odot g + (1 - q) \odot e_g,$$

and use

$$g_{\text{in}}^{\text{GS}}(s, g, q) = [\bar{s}_q, \bar{g}_q, q].$$

Here, $e_s, e_g \in \mathbb{R}^D$ are learned embeddings used for inactive coordinates. The complete, unmasked state $s$ remains available to the learner as the control input; masking is applied only within the goal-comparison interface. We also considered fixed zero and random inactive-coordinate embeddings, but the learned embeddings performed better and are used in all reported GS-HER results.

**Value functions and critics.** Scalar value functions apply a scalar output head to the shared goal-conditioned representation,

$$V_\theta(s, g_{\text{in}}) = \ell(\text{GC-MLP}_\theta(s, g_{\text{in}})).$$

When a backbone requires two value estimates, we instantiate two independently parameterized heads with the same architecture. Critics use the same construction while including the action in the input,

$$Q_\theta(s, a, g_{\text{in}}) = \ell(\text{GC-MLP}_\theta([s, a], g_{\text{in}})).$$

Table 6: Training and evaluation protocol. Unless otherwise stated, backbone-specific algorithmic hyperparameters follow OGBench.

| ITEM | VALUE |
|---|---|
| Implementation framework | PyTorch |
| Algorithmic reference | OGBench JAX implementation |
| GPU | NVIDIA RTX 5090 |
| Minibatch size | 8192 |
| Learning rate | $8 \times 10^{-4}$ |
| Gradient updates | 200 000 |
| Evaluation checkpoints | 160k, 180k, 200k |
| Official test-time goals | 5 |
| Rollouts per goal | 20 |
| Episodes per checkpoint | 100 |
| Episodes per reported seed | 300 |
| Independent training seeds | 5 |
| Method-specific sweeps | None |

**Policies.** Policies use the same goal-conditioned MLP backbone. The actor predicts the mean of a diagonal Gaussian,

$$\mu_\theta(s, g_{\text{in}}) = \text{GC-MLP}_\theta(s, g_{\text{in}}),$$

with

$$\pi_\theta(a \mid s, g_{\text{in}}) = \mathcal{N}\left(\mu_\theta(s, g_{\text{in}}), \sigma^2 I\right).$$

The covariance is fixed before temperature scaling. Evaluation uses the near-deterministic policy configuration of our implementation, so actions correspond to the predicted mean up to negligible sampling noise.

The same architectural components are reused across GCBC, CRL, GCIVL, GCIQL, and GCDL. GS-HER therefore does not introduce a backbone-specific network architecture; it changes the semantics and representation of the relabeled goal supplied to the corresponding base learner.

### G.2 Training and Benchmarking Protocol

Table 6 summarizes the common training and evaluation protocol. Our PyTorch implementation follows the official OGBench algorithms but uses a reduced-compute schedule to make the full cross-backbone comparison feasible on a single GPU.

The original state-based OGBench protocol trains for $10^6$ gradient updates using minibatches of $1024$. To reduce the computational cost of the complete comparison, we increase the minibatch size to $8192$ and shorten training to $2 \times 10^5$ gradient updates. Starting from the OGBench learning rate of $3 \times 10^{-4}$, we apply square-root batch-size scaling,

$$\eta = 3 \times 10^{-4} \sqrt{\frac{8192}{1024}} \approx 8 \times 10^{-4}.$$

A single training run takes approximately $1.5$ hours on one NVIDIA RTX 5090, with minor variation across environments and learning backbones.

The reduced schedule is shared by all goal-semantics variants and makes the full cross-backbone comparison computationally feasible. Importantly, despite using fewer gradient updates, the HER-Full baselines remain broadly comparable to the corresponding results reported by the official OGBench implementations. This provides a useful sanity check that the reduced-compute regime does not produce an obviously undertrained full-state baseline. Our conclusions therefore concern controlled comparisons under this common training budget; we do not establish whether the same quantitative gaps persist asymptotically under substantially longer optimization.

**Official benchmark evaluation.** We evaluate every trained agent after 160k, 180k, and 200k gradient updates. At each checkpoint, the agent is evaluated on each of the five official test-time goals

Table 7: Reference-outcome sampling probabilities used for relabeling. Value and critic updates mix current, future-trajectory, and random-dataset outcomes, whereas actor updates use future outcomes from the same trajectory.

| UPDATE | $\mathbf{p}_{\text{curr}}$ | $\mathbf{p}_{\text{traj}}$ | $\mathbf{p}_{\text{rand}}$ |
|---|---|---|---|
| Value / critic | 0.2 | 0.5 | 0.3 |
| Actor | 0.0 | 1.0 | 0.0 |

using 20 rollouts per goal. Consequently, each per-seed reported score averages

$$3 \text{ checkpoints} \times 5 \text{ goals} \times 20 \text{ rollouts} = 300$$

evaluation episodes. For each seed, success is first averaged across these three checkpoints. Tables and figures then report mean and standard deviation across five independently trained seeds. We use the official OGBench binary goal-success metric for each environment.

### G.3 Data Sampling and Goal Relabeling

All relabeling variants use the same transition and reference-outcome sampling procedure. Given trajectory $\tau^{(n)}$ and timestep $t$, we first sample

$$\left(s_t^{(n)}, a_t^{(n)}, s_{t+1}^{(n)}\right).$$

A reference outcome $g$ is then drawn from one of three sources: the current state, a future state from the same trajectory, or a random state from the offline dataset. The corresponding sampling probabilities are shown in Table 7.

**Current-state outcomes.** With probability $p_{\text{curr}}$, the reference is the current state,

$$g = s_t.$$

**Future-trajectory outcomes.** With probability $p_{\text{traj}}$, the reference is sampled from a future state of the same trajectory. We draw

$$k = \lfloor u^{\alpha} H_{\max} \rfloor + 1, \qquad u \sim \mathcal{U}(0,1), \qquad \alpha = 2,$$

and set

$$t_g = \min(t + k, T), \qquad g = s_{t_g}.$$

This biases sampling toward nearer future outcomes while retaining support for longer temporal offsets. We additionally store the realized temporal distance $k_{\text{traj}} = t_g - t$, which the value-based learners use as a one-sided trajectory-goal reachability witness described below.

**Trajectory-goal demonstration witnesses.** For a future reference $g = s_{t+k_{\text{traj}}}$, the sampled trajectory reaches a member of every query-defined class because $g \in \mathcal{G}(g, q)$. It therefore certifies

$$\tau_{\mathcal{G}(g,q)}(s_t) \le k_{\text{traj}},$$

without asserting that $k_{\text{traj}}$ is the first hitting time.

For GCIVL and GCIQL, define the demonstrated $k$-step value

$$\nu_\gamma(k) = \begin{cases} -k, & \gamma = 1, \\ -\dfrac{1-\gamma^k}{1-\gamma}, & \gamma < 1. \end{cases}$$

For future-trajectory goals, GCIVL lower-bounds each trajectory-derived target entering its expectile update by $\nu_\gamma(k_{\text{traj}})$, while GCIQL lower-bounds its critic target:

$$y \leftarrow \max(y, \nu_\gamma(k_{\text{traj}})), \qquad y_Q \leftarrow \max(y_Q, \nu_\gamma(k_{\text{traj}})).$$

GCDL uses the normalized analogue $y \leftarrow \max(y, -k_{\text{traj}}/H_{\max})$; see Appendix E.1. Since numerically larger values correspond to shorter reachability times, these operations impose a one-sided lower bound on value, equivalently an upper bound on demonstrated hitting time. An earlier query-equivalent state is therefore compatible with a better learned value; GS-HER simply does not relabel that earlier state as terminal.

**Random-dataset outcomes.** With probability $p_{\mathrm{rand}}$, the reference state is sampled uniformly from the complete offline dataset,

$$g = s_{\tilde{t}}^{(\tilde{n})}, \qquad \tilde{n} \sim \mathcal{U}\{1, \ldots, N\}, \qquad \tilde{t} \sim \mathcal{U}\{0, \ldots, T\}.$$

This provides off-trajectory goal supervision to backbones whose objectives use randomly sampled goals.

Reference-outcome sampling, $k_{\mathrm{traj}}$, and the corresponding learner-specific witness rules are shared by HER-Full, HER-Task, and GS-HER. No variant receives projection- or query-specific first-hit labels; GS-HER changes only the goal-conditioning representation after sampling $q$.

### G.4 GS-HER Query Sampling

For each GS-HER relabeled transition, the reference outcome is paired with an independently sampled binary query

$$q \sim p_Q(q), \qquad q \in \{0,1\}^D, \qquad \|q\|_0 \geq 1.$$

The query is sampled independently of both the reference-outcome source and the observed difference between the current and reference states. It specifies which coordinates remain distinguishable in the goal-conditioning representation rather than which properties happened to change along the sampled trajectory.

Our default annotation-free Blockwise variant uses the blockwise query distribution. Each query is drawn as one of three types: a full-state query with probability $0.15$, a random-coordinate query with probability $0.15$, or a blockwise query with probability $0.70$. Full-state queries activate every coordinate. Random-coordinate queries activate an unstructured subset of state coordinates. For a blockwise query, we sample between one and four contiguous coordinate spans independently and activate their union. For each span, its relative length is sampled uniformly from $[0.15, 0.40]$ of the state dimension, and its start location is sampled uniformly among valid indices. Queries with no active coordinates are disallowed.

The GS-HER Semantic variant instead constructs queries from predefined physical state factors such as object position, object orientation, robot pose, and gripper state. This variant uses privileged access to the state factorization and is included as an oracle-query comparison; all primary annotation-free GS-HER results use the blockwise distribution.

### G.5 Model Capacity and Computational Cost

The three relabeling variants use the same backbone type, architectural depth, hidden widths, data, optimization schedule, and update count. Their exact parameter counts are not identical because the dimensionality of the goal-conditioning interface differs. In particular, GS-HER additionally represents the query $q$ and the learned inactive-coordinate embeddings.

For GCIQL, HER-Task, HER-Full, and GS-HER contain respectively $553{,}522$, $599{,}722$, and $628{,}478$ trainable parameters. GS-HER therefore adds $28{,}756$ parameters, or approximately $4.8\%$, relative to HER-Full. GS-HER Blockwise and GS-HER Semantic use the same architecture and parameter count; they differ only in the distribution from which training queries are sampled.

The observed performance differences are therefore not accompanied by changes in network depth, hidden width, training data, optimization schedule, or training budget. We report the exact parameter counts to make the small input-interface difference explicit rather than treating the variants as strictly parameter-matched.

### G.6 Code and Reproducibility

Our implementation will be made publicly available upon de-anonymization. The released code contains the PyTorch implementations of all five base learners, the three goal-semantics interfaces, query generation, OGBench training configurations, and the evaluation protocols used to produce the reported results.