LATENT FUTURE GOAL REPRESENTATIONS

Anonymous authors
Paper under double-blind review

ABSTRACT

Goals in goal-conditioned reinforcement learning are usually single observations, yet abstract tasks are more naturally stated as sets of outcomes, for example reaching either of two states. Such abstract goals can be represented in a latent goal space, but critic and policy have no direct way to use this structure. Typically, they must learn again from value targets what those representations mean, so that an agent trained on single goals, for example, does not generalize to more abstract set-valued goals. We introduce Latent Future Goal Representations (LFG), which learns a shared goal space without critic gradients and trains an action-conditioned predictor to map state-action pairs directly into it. Each prediction represents a latent goal region, and a learned bilinear containment relation scores whether a desired goal belongs to that region. These containment scores serve directly as the critic for policy learning. The predictor is trained on observed future goals, with targets discounted by the time until their arrival. In experiments we find that our shared representation enables the critic to inherit the learned goal space structure, allowing generalization to abstract goals. On OGBench with single observation goals, LFG achieves competitive performance across navigation and manipulation with both state and image observations. Ablations show that relational supervision improves control over temporal supervision alone, while the learned containment relation provides an effective critic readout.

1 INTRODUCTION

Goal-conditioned agents are commonly trained and evaluated on goals that are individual states or images. Many tasks are more naturally stated at a coarser level, such as reaching either of two corners of a maze or bringing a ball to either of two corners of an arena. Such goals are sets of outcomes rather than single outcomes, for example the union of two goals in Figure 1. Hierarchical goal abstractions (HGA) (Wurzberger et al., 2026) make them representable. HGA learns a latent goal space with a containment relation, in which a single state is a singleton goal, a sub-trajectory is a set of states, and a learned energy decides whether one goal is contained in another. Concrete goals, regions and unions of goals are then all elements of one space.

To reach a goal, an agent must relate its actions to the future outcomes they lead to. A goal-conditioned Q-function (Kaelbling, 1993; Schaul et al., 2015) makes this relation explicit: it estimates the return of an action for a given goal. A representation of abstract goals is therefore not yet an agent that pursues them, since the critic must also read those goals correctly. In HGA, as in other methods that learn goal representations with an objective of their own (Myers et al., 2025; Park et al., 2026), the goal representations are handed to a downstream critic as input features. The critic has to recover from value targets what those representations mean. Apart from what the representation itself generalizes, nothing makes a critic trained on single-state goals rate a union of two goals as reachable when either goal is. HGA therefore uses abstract hindsight relabeling, replacing some single-observation goals with observation sequences or abstract goals found by optimization, so that the critic sees abstract goals during training. The structure of the goal space is thus learned twice, once by the representation and again by the critic, and the critic learns it only as far as its training goals cover it. End-to-end critics trained through temporal-difference, contrastive or quasimetric objectives (Kostrikov et al., 2022; Eysenbach et al., 2022; Wang et al., 2023) avoid the second step, but do not explicitly model goal sets.

Sharing the relation between goal space and critic. We instead let the critic use the goal space's relation directly. The future of a state-action pair is the set of states the agent can reach from it, so it

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.

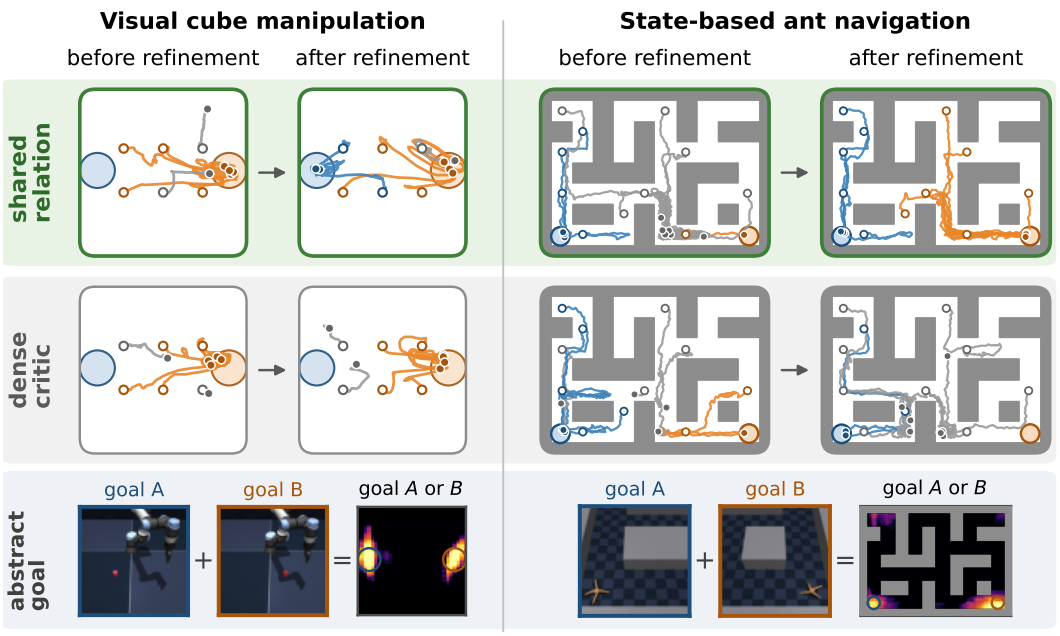


Figure 1: **A critic that shares the goal space's relation allows fine-tuning on abstract goals, while a dense critic on the same goal representations mostly does not.** The agent receives the union of two goals (bottom) on visual cube-single and antmaze-large, and its actor, trained on single goals only, is refined on such goals against the frozen critic. With the shared relation (top) the critic can refine the policy to reach either goal. With a standard dense critic (middle) such a refinement has little effect, and the agent mostly reaches one goal only. Details in Section 4 and Appendix B.7.

is one more set in the goal space: a network can predict it as an element of that space. The critic is then the containment score of the goal in the predicted future (Figure 3). The critic reads every goal, whether a singleton or a set, through the same relation the goal space was trained with. Whatever that relation encodes about sets, for example that a union contains its members, carries over to the critic without abstract goals in the critic's training data. No part of the goal-space structure has to be learned a second time. We introduce Latent Future Goal Representations (LFG), an instantiation of this idea. Its goal space is learned from set and temporal relations, similarly to HGA, and read out with a bilinear containment relation. An action-conditioned predictor is fitted to discounted hindsight targets without bootstrapping or a target network, and the containment scores of its outputs serve as the critic for a behaviour-regularised actor (Fujimoto & Gu, 2021). We test the transfer on union goals, which the goal space represents directly through its union embedding and which none of the agents sees during training (Figure 1). Because the frozen LFG critic already scores these unions, only the actor needs to adapt. Refined against it, the agent reaches either member depending on where it starts. Refined against a dense critic on the same goal representations, the agent improves little. HGA's representations read by a contrastive critic gain more but stay below LFG on every task, and both baselines mostly reach only one member. The shared relation thus passes structure from the goal representation into the policy that separately trained critics do not. We show that the shared critic carries over to unions of goals that were not seen in training, that LFG remains competitive on single-goal OGBench tasks (Park et al., 2025), and that relational supervision improves control over temporal supervision alone (Section 4).

2 METHOD

We study offline goal-conditioned reinforcement learning from a fixed dataset $\mathcal{D}$ of trajectories $\tau = (\boldsymbol{s}_0, \boldsymbol{a}_0, \boldsymbol{s}_1, \boldsymbol{a}_1, \ldots, \boldsymbol{s}_T)$, without further interaction with the environment. We write $\boldsymbol{s} \in \mathcal{S}$ for an observation (a state vector or image) and $\boldsymbol{a} \in \mathcal{A}$ for an action. Training goals are single observations $\boldsymbol{s}_g$. Evaluation includes both single-observation and abstract goals (Section 4). LFG jointly trains a goal space from observations and an action-conditioned predictor into that space. The goal space's containment relation reads out the predictor as a critic, which in turn trains the policy.

2.1 A GOAL SPACE LEARNED FROM SET RELATIONS

Our goal space builds on HGA's set-based representation of goals (Wurzberger et al., 2026): it encodes single-observation goals and abstract goals, such as regions, in a common latent space. An encoder $\phi : \mathcal{S} \to \mathcal{G} = [-1, 1]^k$ places single observations in the space. Every embedded observation is a singleton goal, $\boldsymbol{g} = \phi(\boldsymbol{s}_g)$. For a set of observations X, a permutation-invariant DeepSets (Zaheer et al., 2017) aggregator $\mathrm{OR} : 2^{\mathcal{G}} \to \mathcal{G}$ combines their encodings into one goal representation, $\boldsymbol{g}_X = \mathrm{OR}(\{\phi(\boldsymbol{s}) : \boldsymbol{s} \in X\})$. The same space therefore holds singleton goals, regions and unions.

A learned bilinear relation scores whether one represented goal is contained in another:

$$E(\boldsymbol{x}, \boldsymbol{y}) = \sigma\big(\boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{y}\big), \tag{1}$$

where $E(\boldsymbol{x}, \boldsymbol{y})$ reads as "the set represented by $\boldsymbol{x}$ is contained in the set represented by $\boldsymbol{y}$". A successor operator $\mathrm{SUCC} : \mathcal{G} \to \mathcal{G}$ represents the states that follow a set in the data. It adds temporal direction to the otherwise unordered set representation. All operators act within $\mathcal{G}$, so the same relation E can compare single goals, unions and continuations (Appendix A.2).

We train these components on sampled sub-trajectories (Figure 2). Membership places each observed state inside its sub-trajectory while nesting places a shorter sub-trajectory inside a longer one. Identity makes the union of a single goal equal that goal, while union consistency makes two overlapping sub-trajectories reproduce their whole. The successor objective includes states after a prefix and excludes its earliest states. States and sub-trajectories from other trajectories serve as negatives. To cover both local and distant futures, we sample a fixed number of states from windows whose spans range over two orders of magnitude (Appendix A.2).

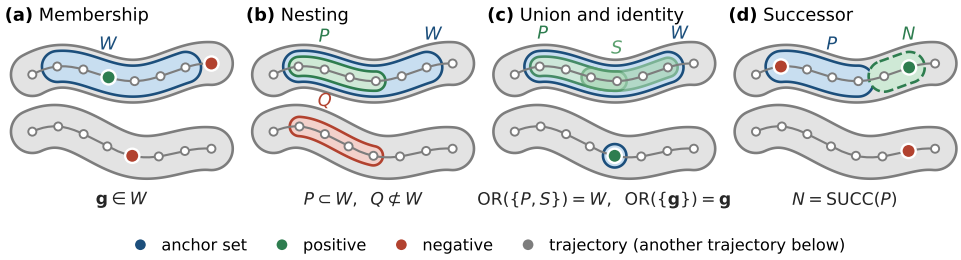


Figure 2: **Four representation objectives shaping the goal space.** Goals are sets of observations that respect the subset relation (a, b), the union property (c) and the successor property (d). Details in Appendix A.2.

2.2 A CRITIC THAT PREDICTS INTO THE GOAL SPACE

An action-conditioned predictor $\psi : \mathcal{S} \times \mathcal{A} \to \mathcal{G}$ maps each state-action pair to a representation of its future outcomes. The containment relation of Equation 1 then scores a goal against this predicted future. We use that score directly as the critic:

$$Q(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}_g) = E\big(\boldsymbol{g}, \psi(\boldsymbol{s}, \boldsymbol{a})\big), \qquad \boldsymbol{g} = \phi(\boldsymbol{s}_g). \tag{2}$$

The critic loss updates only the parameters $\boldsymbol{\theta}$ of ψ. The goal encoder, set operators and relation matrix $\boldsymbol{W}$ receive no critic gradient. Thus the critic uses the goal space's learned relation without retraining it from value targets.

We fit ψ to discounted hindsight targets from the dataset (Figure 3). For a transition $(\boldsymbol{s}_t, \boldsymbol{a}_t)$, a goal observed d steps later on the same trajectory receives target γ^{d-1}. A goal from another trajectory, or one that is not ahead on the same trajectory, receives target zero. These are the returns for a unit reward on arrival, termination at arrival and discount $\gamma \in (0, 1)$:

$$y(\boldsymbol{s}_t, \boldsymbol{s}_g) = \begin{cases} \gamma^{\,d-1} & \text{if } \boldsymbol{s}_g \text{ is sampled at } t+d \text{ on the same trajectory,} \\ 0 & \text{else.} \end{cases} \tag{3}$$

We fit the predictor to these targets over sampled transitions and goals:

$$\mathcal{L}_{\text{critic}}(\boldsymbol{\theta}) = \mathbb{E}_{(\boldsymbol{s}_t, \boldsymbol{a}_t, \boldsymbol{s}_g) \sim \mathcal{D}} \Big[\text{BCE}\big(E(\phi(\boldsymbol{s}_g), \psi_{\boldsymbol{\theta}}(\boldsymbol{s}_t, \boldsymbol{a}_t)),\, y(\boldsymbol{s}_t, \boldsymbol{s}_g)\big) \Big], \tag{4}$$

where BCE denotes binary cross-entropy. No bootstrapping or target network is needed.

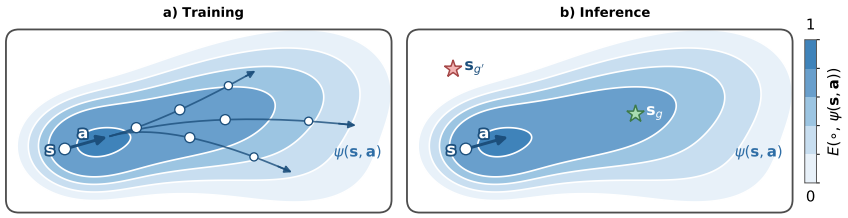


Figure 3: **Learning the latent future.** *(a)* During training the successor set $\psi(\boldsymbol{s}, \boldsymbol{a})$ is grounded in observed futures that are sampled continuations from the data. The predicted future $\psi(\boldsymbol{s}, \boldsymbol{a})$ is fitted through $Q(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}_g) = E(\boldsymbol{g}, \psi(\boldsymbol{s}, \boldsymbol{a}))$ so that a goal reached d steps later scores $\gamma^{\,d-1}$. The dots shrink with the discount. *(b)* During inference actions are evaluated with the same score $Q(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}_g) = E(\boldsymbol{g}, \psi(\boldsymbol{s}, \boldsymbol{a}))$. Unreachable goals $\boldsymbol{s}_{g'}$ score near zero (red star), reachable goals (green star) are contained within the future region $\psi(\boldsymbol{s}, \boldsymbol{a})$.

Finally, a goal-conditioned policy $\pi(\boldsymbol{a} \mid \boldsymbol{s}, \boldsymbol{g})$ is trained with DDPG+BC (Lillicrap et al., 2016; Fujimoto & Gu, 2021), a standard offline actor. It maximises the critic for its own action. Behaviour cloning with coefficient α keeps it close to the dataset actions (Appendix A.1). The shared relation is what makes abstract goals usable. The policy is trained on single goals only. The critic, however, scores any element of the goal space, including abstract goals such as the union of two goals. To adapt the policy to such goals, we keep the goal space, $\boldsymbol{W}$ and ψ fixed. Only the actor is refined, with the unchanged objective. Each transition's goal is replaced by the union of its own future goal with the future goal of another transition in the batch. No value target for the abstract goal is needed, because the frozen critic scores it through the shared relation (Section 4, Appendix B.7).

3 EXPERIMENTS

We test whether the shared containment readout helps the agent pursue abstract goals absent from critic training. We also probe what the goal space and predicted future encode, and evaluate single-observation goals to measure performance on the standard benchmark tasks.

Testing the transfer with abstract goals. We focus on abstract goals, namely union goals, which the goal space represents directly through its OR embedding. A union of two goals is reached when the agent reaches either of them, and no agent sees such a goal during training. We evaluate unions

of two goals on four OGBench tasks. Antmaze-large and antsoccer are learned from states, visual antmaze-medium and visual cube-single from pixels. Each task has five goal pairs (A, B), such as two corners of the maze, two arena corners for the ball, or two cube positions on the table. Each pair is evaluated from a fixed set of starts (Appendix B.7). The agent receives the union of the pair and succeeds when it reaches either goal under the benchmark's criterion. We also report how often the agent reaches the goal of the pair it reaches less often. This shows whether the agent chooses between the two goals or treats the abstract goal as one of them. As a control, we report the success on each goal of a pair on its own. Each agent is first evaluated as trained on single goals, measuring how the complete agent behaves on unions without refinement. Its actor is then refined on abstract goals for 50k updates against its frozen critic (Section 2.2). This tests whether the actor can improve using the critic's existing union-goal scores. The behaviour-cloning term also supplies recorded actions that reach one member of each training union. To separate the effect of the shared relation from that of the goal space, we compare three agents with three seeds each. LFG (shared) is our agent. LFG (dense) uses the same goal space, but its critic is a separately trained dense network that reads the state, the action and the detached goal representation (Appendix B.2.1). It is trained with the same targets and schedule as LFG (shared). In HGA + CRL, the HGA goal representations are the goal input of CRL's critic (Appendix A.3). Both LFG agents receive representations of unions of two goals, and HGA + CRL encodes the goal observations of both goals with its own sequence encoder. All three agents are pre-trained on single-observation goals only, without abstract hindsight relabeling.

Probing the goal space and the predicted future. We place the agent or an object at each position $\boldsymbol{p}$, encode the resulting observation as a goal $\boldsymbol{g}$, and map its containment score without a decoder. The encoder probe $E(\boldsymbol{g}, \phi(\boldsymbol{s}))$ compares each goal with the singleton set of a fixed anchor state $\boldsymbol{s}$, showing the neighbourhood the encoder treats as one state. The critic probe $E(\boldsymbol{g}, \psi(\boldsymbol{s}, \boldsymbol{a}))$ with $\boldsymbol{a} = \pi(\boldsymbol{s}, \boldsymbol{g})$ scores each goal under its corresponding policy action. Because the action changes with the goal, this maps goal-conditioned critic scores rather than the predicted future of one fixed state-action pair (Figure 5, Appendices B.4 and B.6).

Single goals. Sharing the relation constrains the critic: it cannot fit a new function of the goal on top of the goal space. We test ordinary-goal performance on OGBench, namely seven state-based and five image-based environments of maze navigation, antsoccer and object manipulation, each with five evaluation tasks (Park et al., 2025). Following the benchmark, we train for 1M updates on states and 500k on pixels, average the last three evaluations and report three seeds (Appendix A.1). We compare with the end-to-end critics GCIVL, QRL (Wang et al., 2023), CRL (Eysenbach et al., 2022) and GCIQL (Kostrikov et al., 2022), as reported by OGBench. Further we include TRA and DGR which use separately learned representations read by downstream critics, as published by Park et al. (2026). Finally we compare against HGA using CRL for critic learning. Appendix B.1 reruns CRL and GCIQL on a subset of the environments to ensure that our protocol matches.

4 Results

Union goals through the shared relation. Refining the actor on union goals against the frozen critic raises LFG (shared)'s mean success across four tasks from 59 to 74%. The same procedure changes LFG (dense) from 31 to 35% and HGA + CRL from 21 to 38% (Figure 4). The less-used member of each goal pair is reached from 17% of starts by refined LFG (shared), up from 11% for the pre-trained agent. The refined baselines reach that member from 5% and 3% of starts. Even without refinement, LFG (shared) succeeds from 52, 45 and 87% of starts on antmaze-large, antsoccer and visual antmaze-medium, against 34, 6 and 65% for LFG (dense). After refinement, both success and use of the less-used goal improve on every task for LFG (shared). Before refinement all three agents mostly pursue only one of the two goals, and the two baselines still do after refinement. On the navigation tasks, LFG (dense) reaches individual goals from 82 to 95% of starts, so its union-goal weakness is not solely an inability to reach those locations. On visual cube-single it is weak on single goals as well, so that task does not isolate union-goal generalisation. Refining on unions alone also lowers LFG (shared)'s single-goal success from 93 to 87%. Table 1 gives the results by task.

Goal space and latent future goals. The probes show local singleton scores and broader goal-conditioned critic scores (Figure 5). The encoder score is high only within a cell or two of the

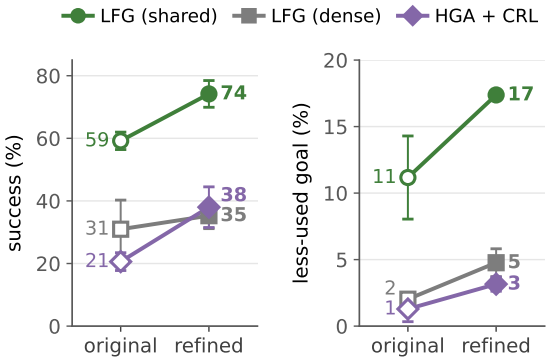


Figure 4: **Performance on abstract goals.** Left: success rate on union goals. Right: share of starts from which the agent reaches the less-used goal of each pair. Abstract goals are created by the union property and given to the policy without fine-tuning (original) and with fine-tuning through the frozen critic (refined). When trained with LFG, the success rate (left) of zero-shot generalization is roughly doubled, compared to baseline critic representations that do not share goal space structure. After fine-tuning the policy with abstract goals, success rates increase and performance becomes more balanced between goals (right).

anchor in the maze or near the cube. The critic score extends along corridors and over the table, beyond the anchor's own representation. Because the action varies with the goal, this map is not the predicted future for one fixed action. On held-out trajectories of four environments, the containment of the state observed d steps ahead decreases monotonically with d (Appendix B.3).

Single goals on OGBench. LFG remains competitive on many single-goal tasks, although its performance varies by environment (Table 2). Among the end-to-end critics, CRL leads navigation, while GCIQL and GCIVL lead antsoccer and manipulation. Among the separately learned representations, HGA's are strongest on the state mazes and weak on every task with an object, and DGR shows the reverse pattern. LFG is close to the strongest end-to-end critic on antmaze-large, humanoidmaze-medium, cube-single and visual antmaze-medium. It has the best result on visual cube-single and the best pixel average. It trails on visual antmaze-large and on the tasks with objects to move, antsoccer and the tasks with several objects, where its goal space does not resolve object positions finely enough (Section 6). The shared readout also matters for single goals. On the long-range antmaze-large tasks, LFG (dense) reaches 54% against 82% for LFG (shared) (Appendix B.2.1). Table 3 shows that both the set and the successor supervision contribute to the performance of LFG. Removing the set losses hurts most on cube-single, from states and from pixels, and on the mazes. Removing the successor term hurts most on antsoccer, antmaze-large and scene.

	antmaze-large		antsoccer		vis. antmaze		vis. cube	
	orig.	ref.	orig.	ref.	orig.	ref.	orig.	ref.
Either goal success (%)								
LFG (shared)	52 ± 12	$\mathbf{78} \pm 4$	45 ± 1	$\mathbf{55} \pm 11$	87 ± 12	$\mathbf{92} \pm 4$	53 ± 14	$\mathbf{72} \pm 7$
LFG (dense)	34 ± 15	29 ± 7	6 ± 2	17 ± 5	65 ± 14	78 ± 3	19 ± 6	17 ± 15
HGA + CRL	16 ± 6	59 ± 3	13 ± 1	24 ± 0	39 ± 13	55 ± 21	14 ± 3	15 ± 1
Single goal success (%)								
LFG (shared)	95 ± 1	$\mathbf{92} \pm 0$	92 ± 3	72 ± 0	98 ± 1	$\mathbf{98} \pm 2$	87 ± 7	$\mathbf{85} \pm 6$
LFG (dense)	82 ± 3	80 ± 3	95 ± 3	$\mathbf{76} \pm 11$	95 ± 0	97 ± 0	37 ± 33	31 ± 29
HGA + CRL	50 ± 10	88 ± 4	56 ± 8	22 ± 1	55 ± 42	57 ± 39	48 ± 7	25 ± 11
Less-used goal ratio (%)								
LFG (shared)	9 ± 2	$\mathbf{16} \pm 8$	6 ± 1	$\mathbf{14} \pm 7$	15 ± 1	$\mathbf{16} \pm 2$	15 ± 8	$\mathbf{24} \pm 4$
LFG (dense)	4 ± 0	2 ± 1	1 ± 1	3 ± 0	3 ± 1	12 ± 1	0 ± 0	3 ± 3
HGA + CRL	0 ± 0	1 ± 0	1 ± 1	5 ± 2	4 ± 4	5 ± 3	0 ± 0	2 ± 1

Table 1: **Abstract goals on four tasks.** Mean over five pairs $\pm$ std over three seeds, before (orig.) and after (ref.) actor refinement. *Either goal success*: the agent receives the union goal and reaches either of its two goals. *Single goal success*: the agent receives each goal of a pair alone. *Less-used goal ratio*: share of starts from which the agent reaches the goal of the pair it reaches less often under the union goal. Bold: highest refined value per block.

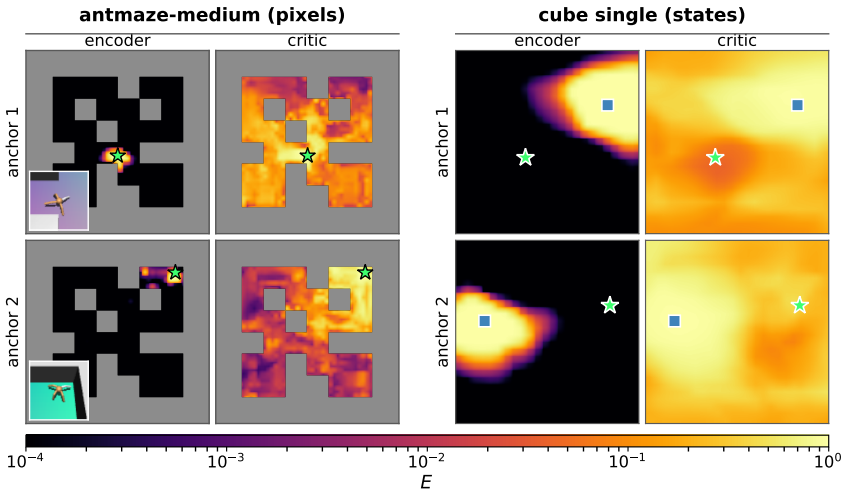


Figure 5: **Encoder and goal-conditioned critic probes** on visual antmaze-medium (left) and state cube-single (right). Each position defines a possible goal $\boldsymbol{g}$. The encoder score $E(\boldsymbol{g}, \phi(\boldsymbol{s}))$ shows that goals are locally active, while the future representation $\psi(\boldsymbol{s}, \pi(\boldsymbol{s}, \boldsymbol{g}))$ extends along corridors or over the table. This is in line with our interpretation that the critic compares the goal to a discounted future region. More examples are in Appendices B.4 and B.5.

Task	GCIVL	QRL	CRL	GCIQL	TRA	DGR	HGA	Ours
antmaze-large	16 ± 5	75 ± 6	83 ± 4	34 ± 4	22 ± 12	28 ± 11	$\mathbf{86 \pm 2}$	82 ± 4
antmaze-giant	0 ± 0	14 ± 3	16 ± 3	0 ± 0	0 ± 0	0 ± 0	$\mathbf{29 \pm 1}$	25 ± 0
humanoidmaze-medium	24 ± 2	21 ± 8	60 ± 4	27 ± 2	21 ± 3	29 ± 3	$\mathbf{62 \pm 2}$	55 ± 2
antsoccer	47 ± 3	8 ± 2	23 ± 2	$\mathbf{50 \pm 2}$	8 ± 2	31 ± 3	16 ± 1	38 ± 4
cube-single	53 ± 4	5 ± 1	19 ± 2	68 ± 6	40 ± 5	$\mathbf{89 \pm 3}$	6 ± 1	68 ± 3
cube-double	36 ± 3	1 ± 0	10 ± 2	40 ± 5	7 ± 2	$\mathbf{60 \pm 4}$	2 ± 1	3 ± 0
scene	42 ± 4	5 ± 1	19 ± 2	51 ± 4	46 ± 6	$\mathbf{72 \pm 6}$	7 ± 1	17 ± 3
visual antmaze-medium	22 ± 2	0 ± 0	$\mathbf{94 \pm 1}$	11 ± 1	48 ± 4	78 ± 4	86 ± 6	91 ± 2
visual antmaze-large	5 ± 1	0 ± 0	$\mathbf{84 \pm 1}$	4 ± 1	13 ± 3	40 ± 4	56 ± 9	47 ± 13
visual cube-single	60 ± 5	41 ± 15	31 ± 15	30 ± 5	31 ± 24	58 ± 5	35 ± 4	$\mathbf{72 \pm 3}$
visual cube-double	$\mathbf{10 \pm 2}$	5 ± 0	2 ± 1	1 ± 1	3 ± 2	9 ± 2	1 ± 0	7 ± 2
visual scene	25 ± 3	10 ± 1	11 ± 2	12 ± 2	15 ± 6	$\mathbf{26 \pm 5}$	11 ± 1	19 ± 1
State average	31	18	33	39	21	**44**	30	41
Pixel average	24	11	44	12	22	42	38	**47**

Table 2: **OGBench success rates (%)** on single goals. Our runs and HGA show mean $\pm$ std over three seeds, while GCIVL, QRL, CRL and GCIQL are as reported by OGBench (eight state and four pixel seeds). Appendix B.1 reruns CRL and GCIQL. TRA and DGR use GCIVL as downstream RL approach as reported by Park et al. (2026). Note that DGR's goal representation is trained with action labels, while TRA, HGA and LFG learn theirs from observations only. Bold font indicates best per row.

States

Task	Full	−Set	−Succ.
antsoccer	$\mathbf{38 \pm 4}$	33 ± 2	31 ± 3
antmaze-large	$\mathbf{82 \pm 4}$	69 ± 3	73 ± 6
cube-single	68 ± 3	36 ± 4	$\mathbf{71 \pm 2}$
scene	17 ± 3	$\mathbf{19 \pm 1}$	9 ± 0
Average	**51**	39	46

Pixels

Task	Full	−Set	−Succ.
visual antmaze-medium	$\mathbf{91 \pm 2}$	78 ± 12	87 ± 2
visual scene	$\mathbf{19 \pm 1}$	16 ± 1	$\mathbf{19 \pm 1}$
visual cube-single	$\mathbf{72 \pm 3}$	5 ± 7	69 ± 1
visual cube-double	7 ± 2	$\mathbf{8 \pm 4}$	7 ± 2
Average	**47**	27	46

Table 3: **Ablations of the goal-space supervision** in state-based (left) and pixel-based (right) environments. Full shows the same success rates as in Table 2 trained with all representation objectives. *−Set* removes all subset objectives, so that the goal space is only trained by the successor loss. *−Succ.* removes the successor loss and keeps all other losses. Success rates are shown as mean ± std over three seeds.

5 Related Work

End-to-end goal representations. Goal-conditioned value functions (Kaelbling, 1993; Schaul et al., 2015) are commonly trained with hindsight relabeling (Andrychowicz et al., 2017), as a supervised policy (Ghosh et al., 2021), by dynamic programming (Kostrikov et al., 2022; Park et al., 2023), by contrastive learning of a critic (Eysenbach et al., 2021; 2022; Zheng et al., 2024), or as a quasimetric (Wang et al., 2023). In all of these the goal representation is learned end-to-end with the critic, through the critic's objective. Like contrastive RL (Eysenbach et al., 2022), our critic $Q(\boldsymbol{s}, \boldsymbol{a}, \boldsymbol{s}_g) = E(\boldsymbol{g}, \psi(\boldsymbol{s}, \boldsymbol{a}))$ of Equation 2 is bilinear in a state-action representation and a goal representation. The two differ in what trains the goal side and in what the score means: contrastive RL trains its goal encoder through the critic loss and reads a density ratio from the binary discrimination of future goals against negatives, whereas LFG learns the goal side by the representation objectives alone, without actions or value targets, and fits the containment score to discounted hindsight labels γ^{d-1}, so the score is graded by time to arrival and needs no bootstrapping.

Separately learned goal representations. A second line of work learns the goal space by a separate objective and the critic on top of it. Temporal representation alignment (Myers et al., 2025) uses temporal contrast and hierarchical goal abstractions (Wurzberger et al., 2026) learn soft set relations, both from observations only. Dual goal representations (Park et al., 2026) shape the goal encoding by temporal distances learned through an auxiliary value objective, so the representation is itself the product of a critic and needs action-labelled data. Each is then consumed by a downstream RL algorithm, which fits its own function of the goal on top of the features. HGA's goal space represents abstract goals such as regions and unions, but its downstream critic and policy receive these representations only as inputs: to make use of abstract goals, both have to be trained on them, which HGA does through abstract hindsight relabeling with representations of observation sequences and goals made more abstract by optimisation in the goal space. LFG learns its goal space separately and from observations only, like TRA and HGA, but removes the downstream value learner: the goal space's own containment relation is the critic, applied to a future predicted into the space, so abstract goals need not appear in the critic's training data (Section 4).

Successor representations and occupancy measures. The successor representation (Dayan, 1993) and successor features (Barreto et al., 2017) represent the future as a feature expectation. Successor states and measures have also been learned as representations (Blier et al., 2021; Touati & Ollivier, 2021; Agarwal et al., 2025), by recursive classification (Eysenbach et al., 2021) and as generative models (Janner et al., 2020), and Myers et al. (2025) use successor features for compositional instructions. Other approaches express future outcomes through density ratios (Eysenbach et al., 2022) or generated samples (Janner et al., 2020). Our predictor $\psi(\boldsymbol{s}, \boldsymbol{a})$ is similar to these successor models, but the future is an element of the goal space and is queried through the learned containment relation E. The relation that tests whether a state belongs to a set therefore also tests whether a goal lies in a predicted future.

Set and order structure in representation learning. Order embeddings (Vendrov et al., 2016), box lattices (Vilnis et al., 2018) and query embeddings on knowledge graphs (Ren et al., 2020) learn partial orders and set operations in a latent space. DeepSets provides permutation-invariant

aggregation (Zaheer et al., 2017). Our predictor makes the predicted future another element of such a goal space.

6 Discussion and Limitations

Shared goal representation. On abstract goals, the shared representation shows better generalization in the zero-shot setting, as well as stronger improvements when used after refining the policy with abstract goals. Fine-tuning only on union goals, however, can sometimes lower single-goal success slightly (c.f. Table 1). Moreover, policies are often biased towards one of the goals initially, where refinement manages to alleviate this bias to some extent. The initial biases are even more pronounced in the baseline methods, where fine-tuning has a very limited effect on bias. We have focused on union goals made of two opposing goals to enforce clear binary decisions in the agent that can be easily analyzed. As most methods are not suited to represent abstract goals, we limit our comparison to HGA and an ablated version of our approach without sharing the goal structure (LFG (dense) in Table 1). Our HGA implementation omits the abstract hindsight relabeling strategy of Wurzberger et al. (2026), so that all critics are trained on the same set of goals for our comparison.

Limits of the learned future and goal space. Like other offline methods, LFG selects actions using a fixed behaviour dataset and is subject to its coverage and distribution shift. The critic fits discounted hindsight targets rather than the actor's own return such that its scores depend on the behaviour data and goal sampling (Appendix B.3). Spatial precision is also limited on tasks with several objects. On cube-double, the goal regions of each cube in Appendix B.5 extend much further across the table than on cube-single, where success requires placing a cube within $4\,\text{cm}$. Persistent scene cues may let cross-trajectory negatives be separated without resolving small within-trajectory displacements. Whether adapting the goal space or drawing negatives within a trajectory improves precise manipulation remains open.

Reusing the representation. The goal encoder, set operators and containment energy use temporally ordered observations without action labels, suggesting they could be pretrained on action-free trajectories. An action-conditioned future predictor and a control policy would still require action-labelled data. Our experiments train the representation and agent concurrently, so separate pretraining remains to be evaluated.

7 Conclusion

Goal-conditioned critics usually learn their goal representation either end-to-end, as a by-product of the critic's objective, or separately, through an objective of its own, whose output a downstream critic receives as input. A separately learned goal space can represent abstract goals such as unions, but a critic trained on single-goal value targets does not automatically inherit that structure. LFG shares the relation between the two. It predicts the future of a state-action pair into the goal space and uses the space's own containment relation as the critic. Policies trained on single goals already reach a substantial part of abstract goals never seen during training. Refining the actor with abstract goals through the frozen shared critic further improves success on every task and mitigates initial biases. On single goals, LFG remains competitive across observation modalities and task families.

AI use statement

Generative AI tools assisted with manuscript drafting and revision, analysis code and visualizations, and auditing experimental reporting. The authors take responsibility for the final text, code and figures.

Ethics statement

This work studies offline reinforcement learning on simulated benchmark tasks. It uses no human data and raises no ethical concerns beyond those generic to robot learning research.

Reproducibility statement

All hyperparameters are listed in Appendix A.1, and the representation-learning procedure and every loss term in Appendix A.2. Datasets and the evaluation protocol are those of OGBench (Park et al., 2025). Table 2 distinguishes our experiments from published baseline results.

References

Siddhant Agarwal, Harshit Sikchi, Peter Stone, and Amy Zhang. Proto successor measure: Representing the behavior space of an RL agent. In *International Conference on Machine Learning*, 2025.

Marcin Andrychowicz, Filip Wolski, Alex Ray, Jonas Schneider, Rachel Fong, Peter Welinder, Bob McGrew, Josh Tobin, Pieter Abbeel, and Wojciech Zaremba. Hindsight experience replay. In *Advances in Neural Information Processing Systems*, 2017.

André Barreto, Will Dabney, Rémi Munos, Jonathan J Hunt, Tom Schaul, Hado van Hasselt, and David Silver. Successor features for transfer in reinforcement learning. In *Advances in Neural Information Processing Systems*, 2017.

Léonard Blier, Corentin Tallec, and Yann Ollivier. Learning successor states and goal-dependent values: A mathematical viewpoint. *arXiv preprint arXiv:2101.07123*, 2021.

Peter Dayan. Improving generalization for temporal difference learning: The successor representation. *Neural Computation*, 5(4):613–624, 1993.

Lasse Espeholt, Hubert Soyer, Remi Munos, Karen Simonyan, Volodymyr Mnih, Tom Ward, Yotam Doron, Vlad Firoiu, Tim Harley, Iain Dunning, Shane Legg, and Koray Kavukcuoglu. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning*, 2018.

Benjamin Eysenbach, Ruslan Salakhutdinov, and Sergey Levine. C-learning: Learning to achieve goals via recursive classification. In *International Conference on Learning Representations*, 2021.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In *Advances in Neural Information Processing Systems*, 2022.

Scott Fujimoto and Shixiang Shane Gu. A minimalist approach to offline reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Manon Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations*, 2021.

Michael Janner, Igor Mordatch, and Sergey Levine. γ-models: Generative temporal difference learning for infinite-horizon prediction. In *Advances in Neural Information Processing Systems*, 2020.

Leslie Pack Kaelbling. Learning to achieve goals. In *International Joint Conference on Artificial Intelligence*, 1993.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *International Conference on Learning Representations*, 2015.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022.

Timothy P. Lillicrap, Jonathan J. Hunt, Alexander Pritzel, Nicolas Heess, Tom Erez, Yuval Tassa, David Silver, and Daan Wierstra. Continuous control with deep reinforcement learning. In *International Conference on Learning Representations*, 2016.

Vivek Myers, Bill Zheng, Anca Dragan, Kuan Fang, and Sergey Levine. Temporal representation alignment: Successor features enable emergent compositionality in robot instruction following. In *Advances in Neural Information Processing Systems*, 2025.

Seohong Park, Dibya Ghosh, Benjamin Eysenbach, and Sergey Levine. HIQL: Offline goal-conditioned RL with latent states as actions. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

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

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

Hongyu Ren, Weihua Hu, and Jure Leskovec. Query2box: Reasoning over knowledge graphs in vector space using box embeddings. In *International Conference on Learning Representations*, 2020.

Tom Schaul, Daniel Horgan, Karol Gregor, and David Silver. Universal value function approximators. In *International Conference on Machine Learning*, pp. 1312–1320, 2015.

Ahmed Touati and Yann Ollivier. Learning one representation to optimize all rewards. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

Ivan Vendrov, Ryan Kiros, Sanja Fidler, and Raquel Urtasun. Order-embeddings of images and language. In *International Conference on Learning Representations*, 2016.

Luke Vilnis, Xiang Li, Shikhar Murty, and Andrew McCallum. Probabilistic embedding of knowledge graphs with box lattice measures. In *Annual Meeting of the Association for Computational Linguistics*, 2018.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *International Conference on Machine Learning*, 2023.

Fabian Wurzberger, Sebastian Gottwald, Zeqiang Zhang, and Daniel Alexander Braun. Hierarchical goal abstractions via learned subset relations. In *Forty-third International Conference on Machine Learning*, 2026.

Manzil Zaheer, Satwik Kottur, Siamak Ravanbakhsh, Barnabas Poczos, Ruslan Salakhutdinov, and Alexander Smola. Deep sets. In *Advances in Neural Information Processing Systems*, 2017.

Chongyi Zheng, Benjamin Eysenbach, Homer Rich Walke, Patrick Yin, Kuan Fang, Ruslan Salakhutdinov, and Sergey Levine. Stabilizing contrastive RL: Techniques for robotic goal reaching from offline data. In *International Conference on Learning Representations*, 2024.

A Implementation details

A.1 Architecture and experimental protocol

Architecture. The goal encoder ϕ maps an observation through two layer-normalized hidden layers of width 256 to a 512-dimensional latent with a $\tanh$ output. The union operator OR is a DeepSets aggregator: a two-layer per-element MLP (width 256, GELU and layer normalization), mean pooling over the members, and a linear projection followed by $\tanh$. The successor operator SUCC is a one-hidden-layer MLP on set representations (width 256, GELU) with a $\tanh$ output, so encoded goals, unions and continuations all lie in the same bounded space. One bilinear logit $\ell(\boldsymbol{x}, \boldsymbol{y}) = \boldsymbol{x}^\top \boldsymbol{W} \boldsymbol{y} / \sqrt{k}$ with $k = 512$ and the score $E = \sigma(\ell)$ define every containment readout, including the critic. The factor $1/\sqrt{k}$ is a fixed rescaling of $\boldsymbol{W}$. The future predictor $\psi(\boldsymbol{s}, \boldsymbol{a})$ is an ensemble of two MLPs (three layer-normalized hidden layers of width 512, $\tanh$ output) on the concatenated state and action, and the critic takes the ensemble minimum.

Following the OGBench implementation, the policy is a goal-conditioned Gaussian MLP (three hidden layers of width 512 without layer normalization, unit standard deviation, unsquashed mean) that receives the encoded goal $\boldsymbol{g}$, the same input as the critic. Actions are clipped to $[-1, 1]$ for the critic query and for execution, and the cloning likelihood uses the unsquashed Gaussian. On pixels every network that reads an observation has its own IMPALA-style CNN (Espeholt et al., 2018) in front of its MLP, and goals pass through the goal encoder's CNN.

Training. Table 4 lists every setting. The representation, critic and policy losses are summed and optimized jointly with Adam (Kingma & Ba, 2015). A critic goal is the next state with probability 0.2, a state of the same trajectory at an offset $d \sim \mathrm{Geom}(1 - \gamma)$, clipped at the trajectory end, with probability 0.5, and a random dataset state otherwise. Its target is decided by the sampled index: a goal that lies d steps ahead on the same trajectory has target $\gamma^{\max(d-1,0)}$, also when the random branch drew it, and every other goal, including the trajectory's own past, has target zero. Policy goals are future states of the same trajectory, drawn uniformly in time. The actor objective is DDPG+BC. It maximizes the containment logit of its action under the minimum over the two critic heads plus α times the log-likelihood of the dataset action.

Evaluation. Following OGBench, reported scores average the evaluations at 800k, 900k and 1M updates for states and at 300k, 400k and 500k for pixels. Each evaluation runs the deterministic policy for 50 episodes on each of the five task goals.

A.2 Representation objectives

Sampling. Each representation batch contains 256 windows for states or 32 for pixels. We sample windows in pairs from the same trajectory, choosing their positions independently. Their lengths range from 16 to 256 time steps, with short and long windows sampled on a log scale. A window cannot exceed its trajectory. From each window we select 16 states, one at random from each of 16 roughly equal portions, so every window has the same number of members. The union operator OR encodes these members as a whole window W_i and also encodes overlapping early and late portions as its prefix P_i and suffix S_i. The successor operator predicts a continuation $N_i = \mathrm{SUCC}(P_i)$ from the prefix (Figure 2).

Balanced cross-entropy. For a relation "$\boldsymbol{x}_i$ is contained in $\boldsymbol{y}_j$" the batch gives the matrix $E_{ij} = E(\boldsymbol{x}_i, \boldsymbol{y}_j)$. With $\mathcal{O}$ the pairs (i, j) whose rows come from different trajectories, the balanced cross-entropy

$$\mathcal{L}_{\mathrm{bal}}(E) = \frac{1}{B} \sum_i \mathrm{BCE}(E_{ii}, 1) + \frac{1}{|\mathcal{O}|} \sum_{(i,j) \in \mathcal{O}} \mathrm{BCE}(E_{ij}, 0)$$

weights the positives and the cross-trajectory negatives equally, regardless of their counts. The binary cross-entropy is computed from the logit for numerical stability.

	state-based	pixels
Gradient steps	1M	500k
Batch size B	1024	256
Optimizer / learning rate	Adam / $3 \cdot 10^{-4}$	Adam / $3 \cdot 10^{-4}$
Discount γ	0.99 / 0.995 (see caption)	0.99
BC coefficient α	0.3 locomotion / 3.0 manipulation	0.3 locomotion / 3.0 manipulation
Latent dimension k	512	512
Goal-encoder MLP	(256, 256)	CNN + (256, 256)
Operator hidden width (OR, SUCC)	256	256
ψ / policy MLP	(512, 512, 512)	CNN + (512, 512, 512)
ψ ensemble size	2 (min)	2 (min)
Image encoder	–	`impala_small`
Random-crop probability	0	0.5 scene / 0 else
Representation rows B_{rep}	256	32
Members per window K	16	16
Window length L	$\text{logU}[16, 256]$	$\text{logU}[16, 256]$
Representation loss weights	1 (all terms)	1 (all terms)
Critic goal mixture (next / trajectory / random)	0.2 / 0.5 / 0.3	0.2 / 0.5 / 0.3
Policy goal mixture (trajectory)	1.0	1.0
Reporting evaluation interval	100k	100k
Episodes per task	50	50

Table 4: **Hyperparameters.** One configuration per observation setting, except for the BC coefficient α and the discount γ, which is 0.995 on humanoidmaze-medium and antmaze-giant and 0.99 otherwise.

Loss terms. Let $\boldsymbol{m}_i$ be a randomly chosen member of window i, $\boldsymbol{x}_i$ its first member, $\boldsymbol{o}_i$ a later member outside the prefix and $\boldsymbol{p}_i$ an early member before the suffix begins. The representation loss is the sum of five terms with weight one,

$$\begin{aligned}
\mathcal{L}_{\text{mem}} &= \mathcal{L}_{\text{bal}}\big(E(\boldsymbol{m}_i, W_j)\big) + \mathcal{L}_{\text{bal}}\big(E(\boldsymbol{m}_i, \boldsymbol{m}_j)\big), \\
\mathcal{L}_{\text{nest}} &= \mathcal{L}_{\text{bal}}\big(E(P_i, W_j)\big) + \mathcal{L}_{\text{bal}}\big(E(S_i, W_j)\big), \\
\mathcal{L}_{\text{union}} &= \tfrac{1}{k}\,\|\text{OR}(\{P_i, S_i\}) - W_i\|^2 + \tfrac{1}{k}\,\|\text{OR}(\{\boldsymbol{x}_i\}) - \boldsymbol{x}_i\|^2, \\
\mathcal{L}_{\text{succ}} &= \mathcal{L}_{\text{bal}}\big(E(\boldsymbol{o}_i, N_j)\big) + \text{BCE}\big(E(\boldsymbol{p}_i, N_i), 0\big), \\
\mathcal{L}_{\text{pair}} &= \text{BCE}\big(E(\boldsymbol{m}_i, W_{i'}),\, y_i\big),
\end{aligned}$$

with squared errors averaged over the batch. The membership term places each member inside its own window and inside its own singleton, and outside the windows and members of other trajectories. The nesting term does the same for the prefix and the suffix. The union term makes the union of the two overlapping sub-windows reproduce the window, and it ties the singleton of a state to the state's encoding. The successor term places the later members of a window inside the continuation of its prefix, and it excludes the first members and the states of other trajectories. Its labels come from window positions only, so no discount enters the representation. The paired-window term scores a member against its paired window i' from the same trajectory, with label $y_i = 1$ if the member lies inside that window and $y_i = 0$ otherwise. These are the relations of Figure 2, with self-containment grouped under membership and the paired-window term added.

Design choices relative to hierarchical goal abstractions. HGA learns soft set relations with discrete goal representations and recurrent aggregation over contiguous trajectory segments, and hands them to a separate value learner. LFG keeps the relational supervision and changes three properties so that predicted futures can be elements of the space. Its representations are continuous and bounded, so singleton goals, unions and predicted futures share one geometry. Its aggregation is permutation-invariant (Zaheer et al., 2017), so a set does not depend on the order of its members, and temporal direction enters only through SUCC. Its windows have log-uniform lengths with a

fixed member count, which exposes the space to short and long collections at constant cost. The central difference is the role of the relation: containment reads values from the action-conditioned future instead of supplying goals to another learner.

A.3 HGA BASELINE

The HGA column of Table 2 uses the discrete variant of hierarchical goal abstractions (Wurzberger et al., 2026) as the goal representation of OGBench's CRL agent, following the reference implementation. A one-layer GRU over a contiguous trajectory window produces 32 categoricals of 16 classes (512 dimensions, matching our latent), sampled as straight-through one-hot vectors. A two-layer MLP on the concatenation of two representations is the subset relation. The objectives are nesting, per-frame membership, unions of two windows, bottom and top elements, batch-shuffled cross-trajectory negatives and reversed positives, with a KL to the uniform categorical prior under a free-bits floor warmed up over the first 10% of training. Windows have a length uniform in $[6, 49]$, and sub-windows at most a third of it. Each representation update uses 256 windows and happens once per 16 agent updates, as in the original schedule, at learning rate 10^{-3}. On pixels the frames pass through the benchmark's IMPALA encoder, and a Gaussian image decoder reconstructs one frame of the window from its representation during the first 20% of training. The CRL critic and actor receive the sampled representation of the single goal observation and never update it. All other settings are the benchmark defaults.

B ADDITIONAL EXPERIMENTS

B.1 BASELINE RERUNS

Table 5 reruns CRL and GCIQL with the OGBench reference implementations on three state environments, three seeds each, under our evaluation protocol. The rerun means lie within one published standard deviation for CRL on all three environments and for GCIQL on antsoccer. GCIQL reruns higher than published on antmaze-large and cube-single. On cube-single its rerun mean exceeds ours, while the published mean equals it. We keep the published numbers in Table 2 for comparability with the rest of the benchmark.

	CRL		GCIQL		LFG
Task	published	rerun	published	rerun	(ours)
antmaze-large	83 ± 4	81 ± 4	34 ± 4	39 ± 4	82 ± 4
antsoccer	23 ± 2	22 ± 1	50 ± 2	50 ± 1	38 ± 4
cube-single	19 ± 2	20 ± 1	68 ± 6	74 ± 3	68 ± 3

Table 5: **Baseline reruns.** Success rates (%) of CRL and GCIQL as published in OGBench (eight seeds) and rerun with our protocol (three seeds, mean $\pm$ std), next to LFG (three seeds).

B.2 ABLATIONS

Table 3 in Section 4 ablates the set and successor supervision of the goal space. This section compares the shared readout with a dense critic on the same goal space and the relational objective with a temporal-contrastive one.

B.2.1 LFG (SHARED) VERSUS LFG (DENSE)

LFG (dense) replaces the shared readout $E(\boldsymbol{g}, \psi(\boldsymbol{s}, \boldsymbol{a}))$ with a separately trained critic on the same goal features, a scalar MLP on the state, the action and the singleton goal representation. Critic gradients do not train the goal representation in either agent. The dense models use the main representation configuration and training budget with three seeds, and they are the dense models of the union-goal experiments (Appendix B.7). Table 6 shows LFG (shared) far ahead on antmaze-large, cube-single and visual cube-single. On visual cube-single, one LFG (dense) seed collapses. The two are close on visual antmaze-medium, antsoccer and cube-double. Since both agents read the same goal features, the large gaps come from the readout.

Task	LFG (shared)	LFG (dense)
antmaze-large	$\mathbf{82 \pm 4}$	54 ± 5
antsoccer	38 ± 4	$\mathbf{41 \pm 2}$
cube-single	$\mathbf{68 \pm 3}$	53 ± 2
cube-double	3 ± 0	$\mathbf{4 \pm 1}$
visual antmaze-medium	$\mathbf{91 \pm 2}$	$\mathbf{91 \pm 1}$
visual cube-single	$\mathbf{72 \pm 3}$	43 ± 30

Table 6: **LFG (shared) and LFG (dense) on the same goal features.** Success rates (%), mean $\pm$ std over three seeds. Bold: higher mean.

B.2.2 Temporal-contrastive representation objective

We replace the relational representation objective with the temporal-alignment loss of TRA (Myers et al., 2025), following the implementation released with dual goal representations (Park et al., 2026). This changes the objective within LFG and does not evaluate the complete TRA or DGR agent. Both variants keep the singleton representation $z(\boldsymbol{s}) = \mathrm{OR}(\{\phi(\boldsymbol{s})\})$ and the shared matrix $\boldsymbol{W}$, and the TRA variant drops all set, identity and successor losses. For N pairs of a current and a future observation, the future offsets are drawn geometrically with $\gamma = 0.99$ and clipped at the episode end. With $L_{ij} = z(\boldsymbol{s}_j^+)^\top \boldsymbol{W} z(\boldsymbol{s}_i)/\sqrt{k}$, the loss is

$$\mathcal{L}_{\mathrm{TRA}} = -\frac{1}{N^2} \sum_{i=1}^{N} \left[\log \frac{\exp L_{ii}}{\sum_j \exp L_{ij}} + \log \frac{\exp L_{ii}}{\sum_j \exp L_{ji}} \right]. \tag{5}$$

The batch sizes ($N = 256$ for states and 32 for pixels), the learning rate, the architecture, the critic, the policy objective and all other settings are unchanged. Like the relational objective, this loss uses neither actions nor rewards.

Task	LFG	TRA objective
antmaze-large	$\mathbf{82 \pm 4}$	$\mathbf{82 \pm 1}$
antsoccer	$\mathbf{39 \pm 4}$	12 ± 0
cube-single	$\mathbf{69 \pm 3}$	29 ± 1
cube-double	$\mathbf{3 \pm 0}$	2 ± 1
visual antmaze-medium	$\mathbf{91 \pm 2}$	87 ± 1
visual cube-single	$\mathbf{72 \pm 3}$	0 ± 0

Table 7: **Relational versus temporal-contrastive goal-space supervision.** Success rate (%), mean $\pm$ std over three matched seeds. Bold: higher mean. Critic and actor are unchanged.

Table 7 shows large gaps in favour of relational supervision on cube-single and antsoccer, and on visual cube-single the temporal-contrastive objective loses almost all success. The gap is smaller on visual antmaze-medium, the two are equal on antmaze-large, and both are near zero on cube-double. Together with the $-$Set ablation, this indicates that relational supervision provides distinctions for control that this temporal objective does not.

B.3 Containment consistency of the predicted future

We test whether the predicted future orders goals by when they are reached, on held-out OGBench validation trajectories, with the main models of antmaze-large, antsoccer, cube-single and visual cube-single (one seed each, 1024 to 2048 anchor transitions). For each anchor $(\boldsymbol{s}_t, \boldsymbol{a}_t)$ and offset $d \in \{1, \ldots, 256\}$ we score the state observed d steps later, $E(\phi(\boldsymbol{s}_{t+d}), \psi(\boldsymbol{s}_t, \boldsymbol{a}_t))$, with the head minimum used by the critic. The score decreases monotonically with d on all four environments, and a goal from another trajectory scores low (Figure 6). At long horizons the profile lies below the discount curve. This is expected, since the training target averages the label of a reached goal with the zero labels of similar goals that are not reached, so the score is ordered like the discounted arrival but not calibrated to it.

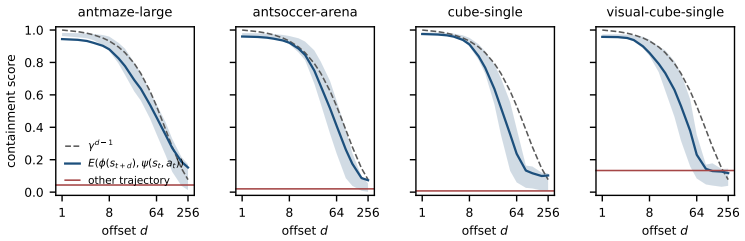


Figure 6: **Containment on held-out trajectories** of the state observed d steps ahead in $\psi(\boldsymbol{s}_t, \boldsymbol{a}_t)$ (band: interquartile range over anchors), with the discount curve and the other-trajectory floor.

B.4 Value maps and direction of motion on the mazes

Value maps on the mazes. Figures 7 and 8 show the two probes of Section 3 for three anchors each, on antmaze-large from states and visual antmaze-medium from pixels. On states a goal moves the agent's xy in the anchor state, and on pixels it is an observation rendered with the anchor's pose at that position. The encoder probe stays within one or two cells of the anchor, while the critic probe extends along the corridors, so the spatial extent of the value comes from ψ and not from the encoder.

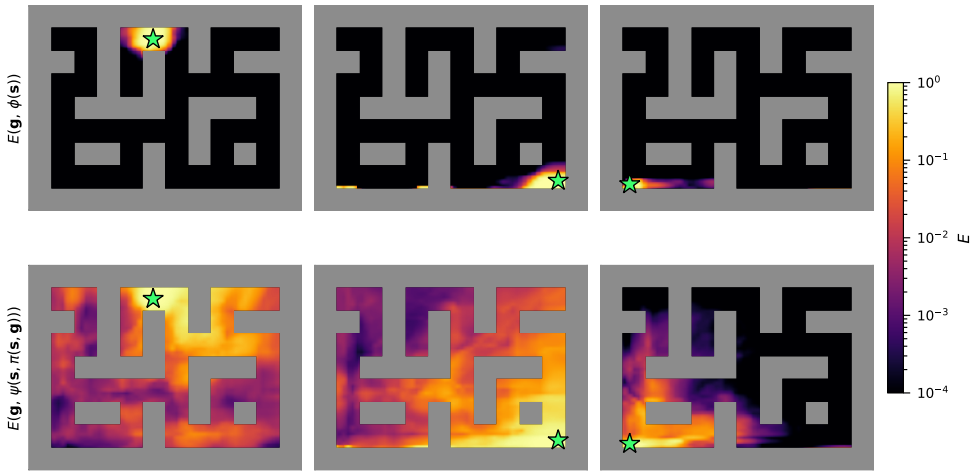


Figure 7: **Encoder probe (top) and critic probe (bottom) on antmaze-large (states)** for three anchors (star), with walls in grey.

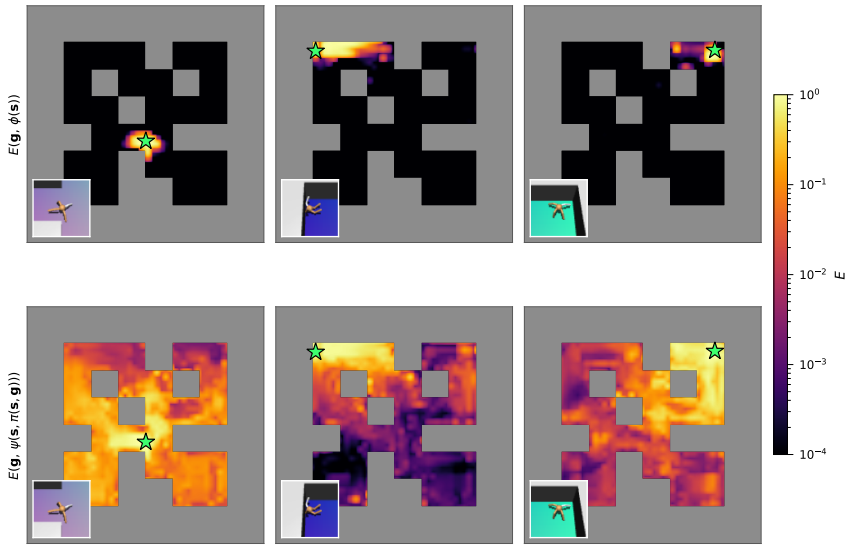


Figure 8: **Encoder probe (top) and critic probe (bottom) on visual antmaze-medium (pixels)**, with goals rendered as 64×64 observations. The inset shows the anchor frame.

Direction of motion. Figure 9 tests whether the critic probe depends on the direction of motion and not only on position. For an example anchor we take the dataset states within one maze unit of its position whose ten-step displacement points along the anchor's direction rotated by 0, 90, 180 and 270 degrees, each with its own recorded action. Although the four states are at nearly the same position, the high-scoring region changes with their direction of motion, from states and from pixels.

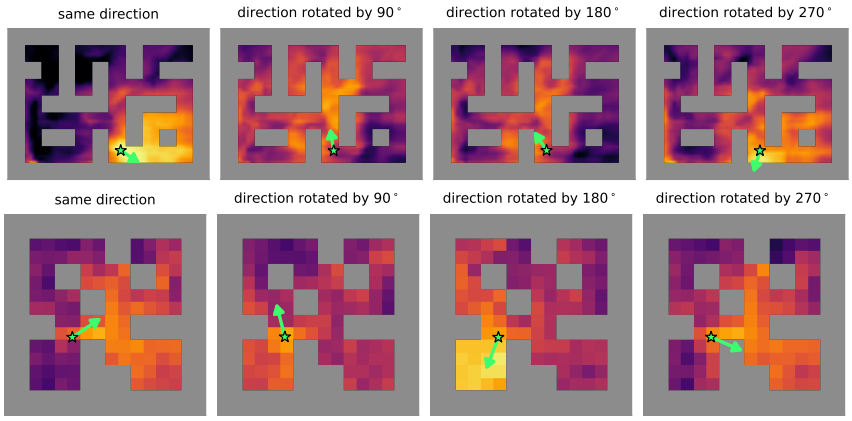


Figure 9: **Same position, different direction of motion.** Critic probe for four dataset states near the anchor position whose ten-step displacement (arrow) is rotated by 0, 90, 180 and 270 degrees, on antmaze-large (top) and visual antmaze-medium (bottom). Each map is min-max scaled in $\log E$.

B.5 Probe maps on the cube tasks

Figures 10 and 11 apply the two probes to the cube-single and cube-double models from pixels and from states. The anchor is a dataset state with the cubes on the table and the effector at least 12 cm from every cube. Each goal moves one entity to a position on a grid over the table: the effector, placed by inverse kinematics at the anchor's height and yaw, or one cube at its resting height and orientation. On pixels the goal is rendered.

Goals that move a cube are contained in the anchor's singleton set only near that cube, and in cube-double each cube's map is centred on that cube alone. The extent of this region differs between

the tasks. On cube-single it is compact, while on cube-double it extends much further across the table, which is consistent with the poor cube-double control (Section 6). Goals that move only the effector are contained almost everywhere on the table in all four models. The goal space thus barely distinguishes where the gripper is, which suits goals defined by object positions. The critic probe shows the same structure, spread more broadly.

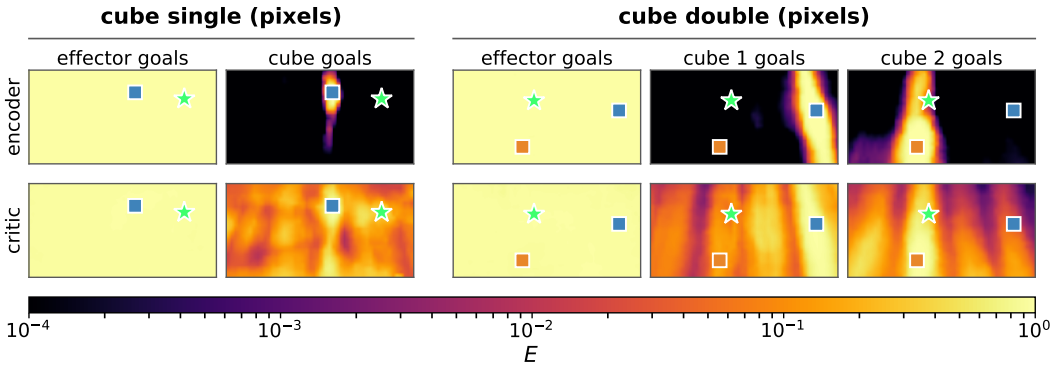


Figure 10: **Encoder probe (top) and critic probe (bottom) on the cube tasks from pixels.** The table seen from above (y horizontal, x vertical), where each pixel is a goal with one entity (the effector or one cube) moved to that position. Star: effector, squares: cubes. All maps share one logarithmic colour scale.

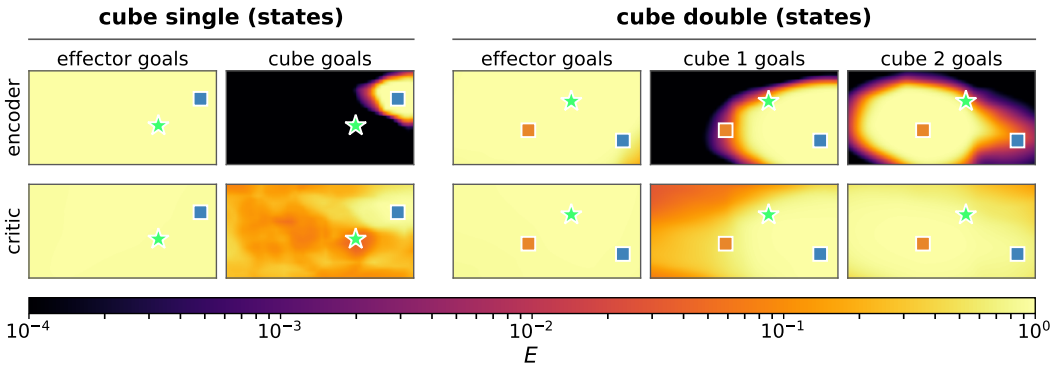


Figure 11: **Encoder probe (top) and critic probe (bottom) on the cube tasks from states**, as in Figure 10, with the edited simulator state as goal.

B.6 Agent and ball on antsoccer

Figure 12 applies both probes on antsoccer to goals that move either the agent or the ball. The encoder probe keeps either edit within a few units of the moved object, so the encoder resolves both objects. The critic probe spreads these sets into a broad basin for agent goals and into tighter corridors for ball goals. Figure 13 places the agent beside a resting ball at four bearings. The high-value region around the ball changes with the agent's placement, so the critic reads the configuration of agent and ball and not the ball position alone. A consistent preference for the push side is not visible.

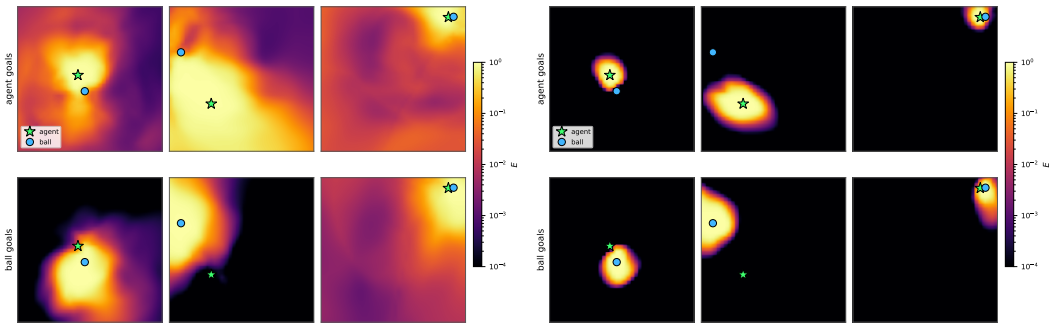


Figure 12: **Agent and ball goals on antsoccer-arena.** Each pixel is a goal obtained by editing a dataset state (markers: agent and ball), and columns are three anchors. Top rows move the agent, bottom rows the ball. Left: critic probe (log scale). Right: encoder probe.

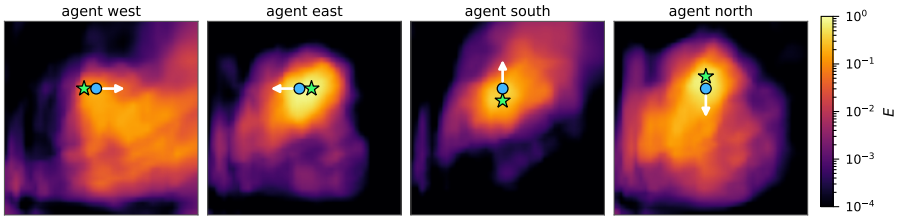


Figure 13: **Value around agent-ball contact.** The agent is placed 1.5 units beside a resting ball at four bearings (arrow: push direction). The maps show the critic probe as the goal ball position varies, with the goal's agent position set to the ball's.

B.7 Abstract goals: refining the actor on unions

Protocol. For each task and seed we take the agent trained on single goals and freeze the goal encoder, both operators, the relation matrix and the critic, and the evaluation script verifies that these parameters are unchanged. The critic is ψ read through the relation for LFG (shared) and the scalar MLP of Appendix B.2.1 for LFG (dense). Only the actor is refined, for 50k updates with the unchanged DDPG+BC objective and the task's α, on union goals. For every transition the goal is $\mathrm{OR}(\{\phi(\boldsymbol{s}_g), \phi(\boldsymbol{s}'_g)\})$, where $\boldsymbol{s}_g$ is the transition's own future goal and $\boldsymbol{s}'_g$ the future goal of another transition of the batch. The behaviour-cloning target remains the recorded action, which leads to $\boldsymbol{s}_g$ and hence into the union. HGA + CRL refines its actor in the same way against its frozen CRL critic, on its own union representation, the sequence representation of the merged window over both goals.

Goal pairs. Each task has five goal pairs, and each agent is evaluated with three seeds. On antmaze-large and visual antmaze-medium a corner goal is a set of 32 dataset states within 1.7 m of a maze corner, and the union goal is OR over the encodings of all 64 states of the two corners. Single-goal controls use the 32 states of one corner. The five pairs are the left, right, bottom, top and diagonal corners. They are evaluated from all free maze cells outside the goal cells (44 and 24 starts), and a corner counts as reached within 1 m of its mean position. On antsoccer the goal is the ball at one of two arena corners, with the same five pairs, from 15 ball starts, and a corner counts as reached when the ball is within 2 m. On visual cube-single the two cube targets are rendered as goal images from the same scene, 16 to 44 cm apart along y, along x or on a diagonal, with 12 to 20 starts between them, and a target counts as reached when the cube is within 4 cm of it.

Results. Table 1 lists the results per task, and Section 4 discusses them. Figures 18–21 show the rollouts of every goal pair, and the antmaze-large panel of Figure 1 shows the bottom pair. Figures 14–17 show what the union goals look like in each agent's goal space. Scored against goals at every position, the union of both LFG agents contains a region around each of the two goals and little in between, while HGA's union representation mostly covers one of the goals or the space between them.

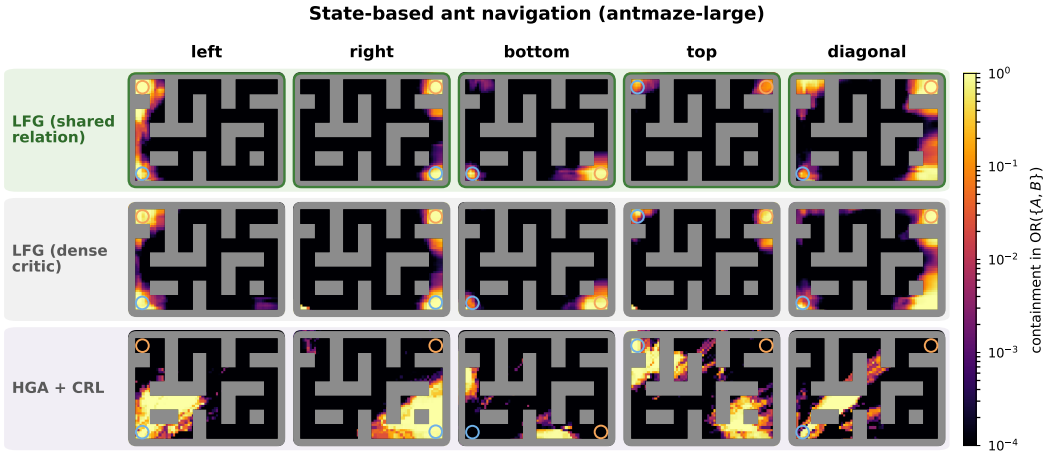


Figure 14: **Union goals in each agent's goal space, state-based ant navigation.** Containment of a goal at each position in the union goal $\mathrm{OR}(\{A, B\})$ the agent receives (log scale, maximum over body poses, circles: the two goals), for five goal pairs and three agents.

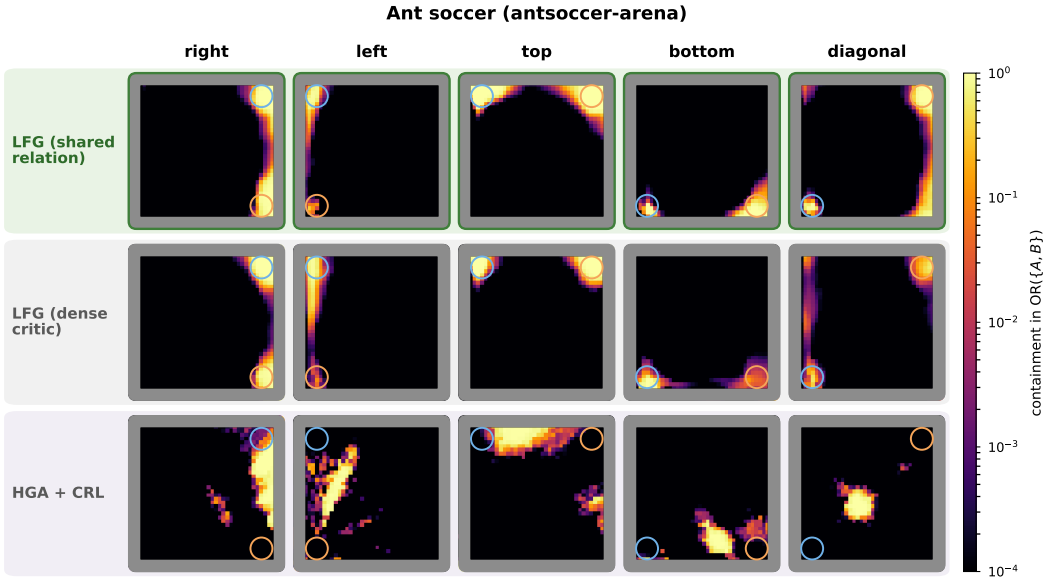


Figure 15: **Union goals in each agent's goal space, ant soccer**, as in Figure 14.

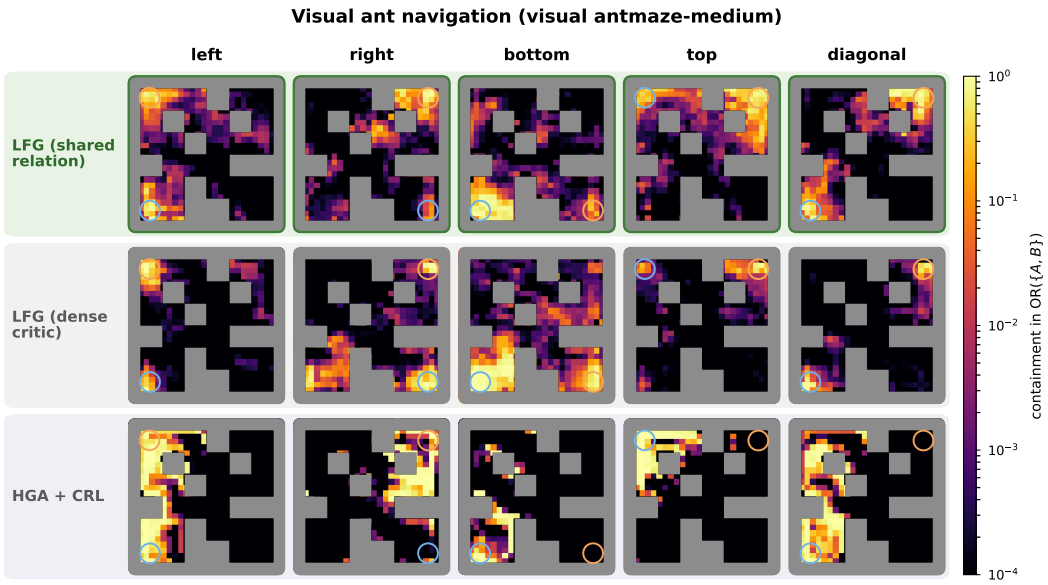


Figure 16: **Union goals in each agent's goal space, visual ant navigation**, as in Figure 14.

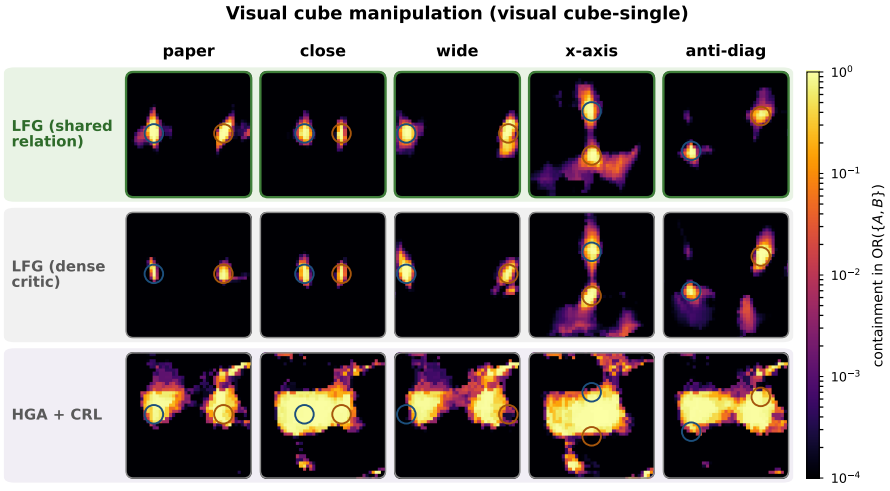


Figure 17: **Union goals in each agent's goal space, visual cube manipulation**, as in Figure 14.

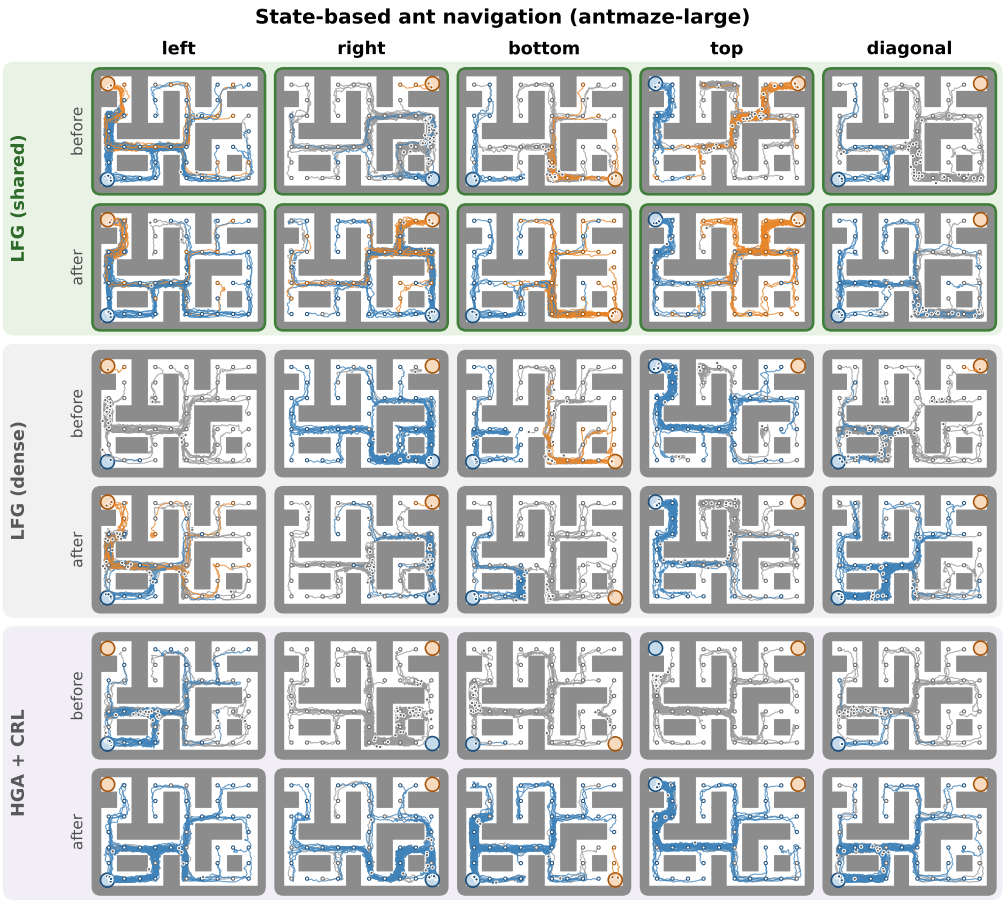


Figure 18: **Union-goal rollouts, state-based ant navigation** (44 starts per pair). Rows are agents before and after refinement, columns are goal pairs. Paths stop at the first goal reached (blue: A, orange: B, grey: neither). Per pair and agent the seed with the largest change is shown.

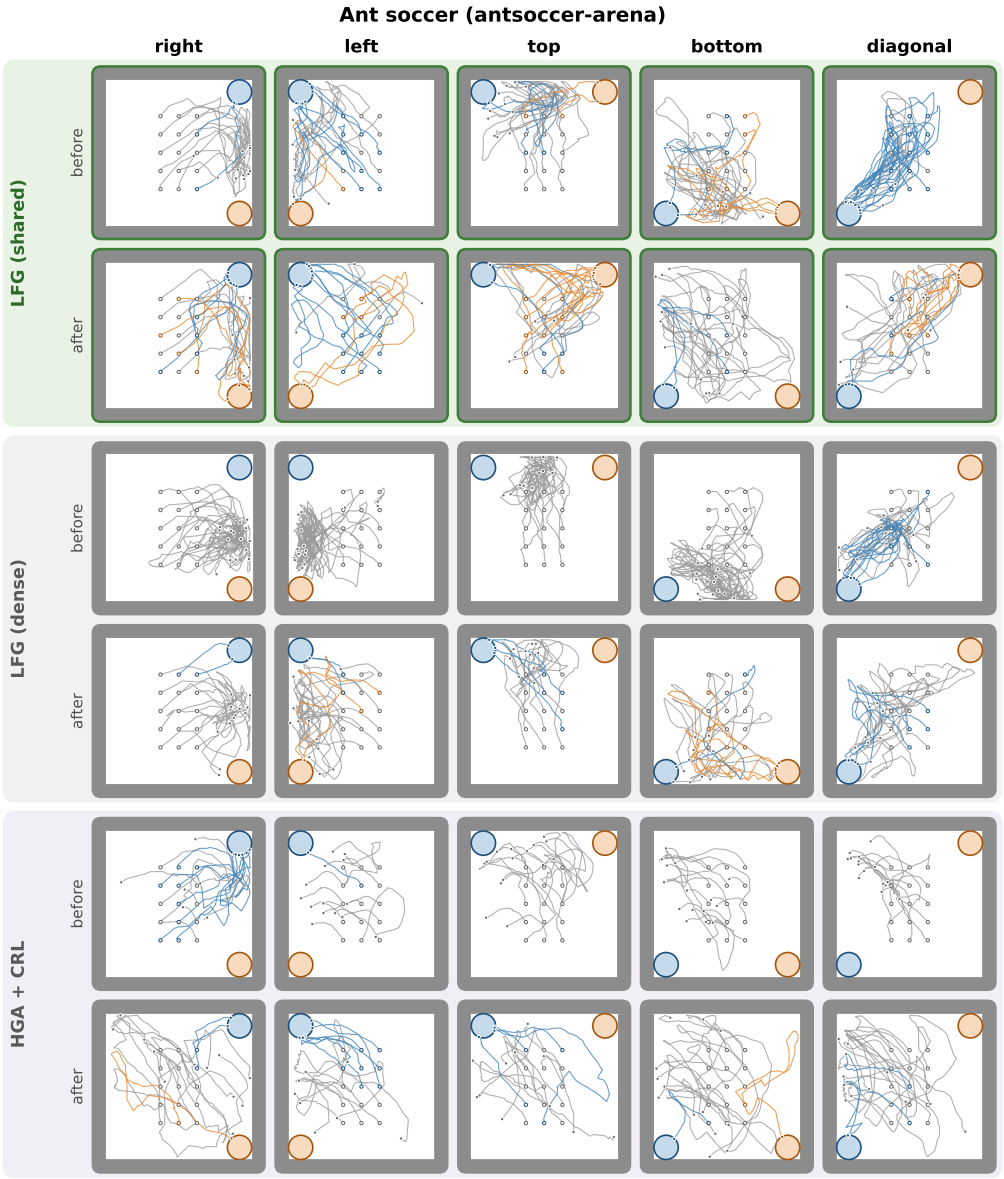


Figure 19: **Union-goal rollouts, ant soccer** (ball paths, 15 starts per pair), as in Figure 18.

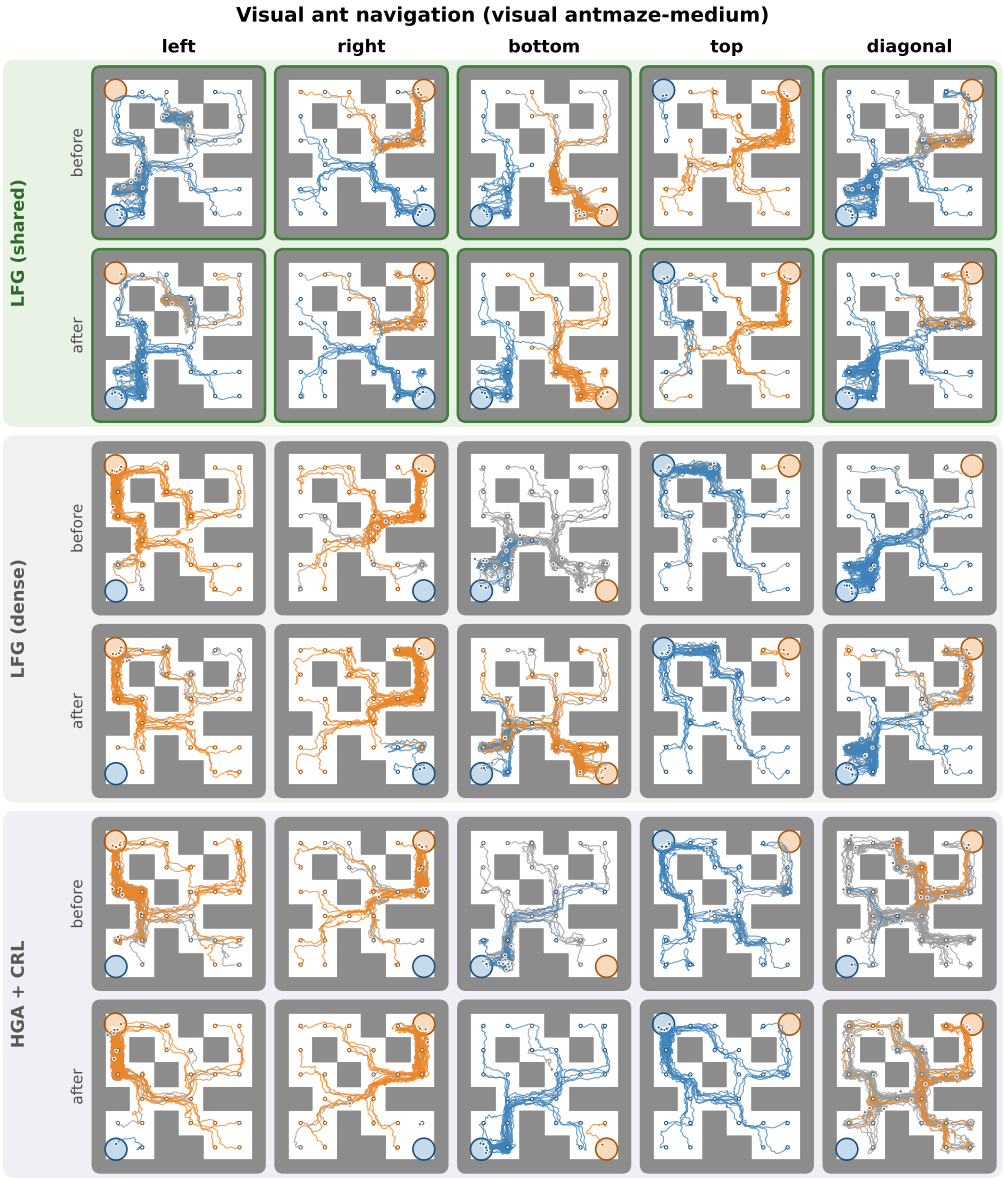


Figure 20: **Union-goal rollouts, visual ant navigation** (24 starts per pair), as in Figure 18.

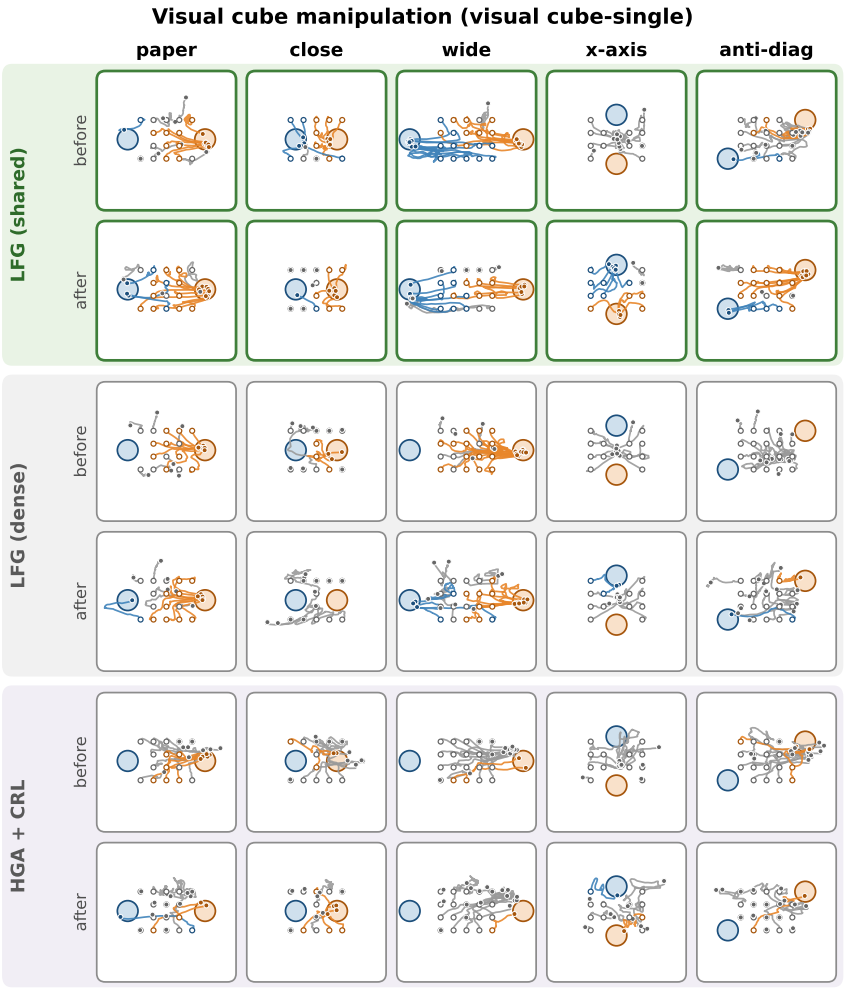


Figure 21: **Union-goal rollouts, visual cube manipulation** (12 to 20 starts), as in Figure 18.