FROM SEARCH TO POLICY: MODEL-BASED IMPROVEMENT FOR OFFLINE VISUAL CONTROL

Anonymous authors
Paper under double-blind review

ABSTRACT

LeWorldModel (LeWM) provides predictive visual representations as a foundation for goal-directed control. Temporal reachability gives these representations a geometry based on the interaction cost of reaching one state from another. We introduce Directed Temporal Representations for Control (DTRC) to learn this temporal geometry from offline trajectories. Using temporal supervision from offline trajectories, we learn a control representation and a directed temporal distance on frozen LeWM features. We show that learning temporal reachability aligns latent distance with goal-reaching cost and turns distance reduction into a direct measure of action progress. We demonstrate empirically that this temporal geometry leads to higher success rates across a diverse suite of visual goal-conditioned control tasks.

1 INTRODUCTION

Latent world models compress visual observations into predictive representations and learn action-conditioned dynamics in latent space, enabling control through imagined rollouts (Hafner et al., 2019; 2020; Hansen et al., 2022; 2024). Visual goal-reaching planners evaluate candidate actions through distances between predicted latent states and goal representations (Zhou et al., 2025; Maes et al., 2026). Latent geometry therefore becomes part of the control objective. Recent work shows that shaping this geometry can align Euclidean distance with feasible transition structure and improve planning (Wang et al., 2026). Temporal reachability provides a different organization based on goal-reaching interaction cost. Latent proximity does not determine this cost, and the cost may differ across opposite transition directions (Wang et al., 2023). A control-oriented representation should organize predictive features according to temporal reachability rather than visual proximity alone.

Direct goal-conditioned policies show that pretrained visual features can support action prediction without iterative test-time search (Nguyen et al., 2026). Recorded trajectories provide an additional source of control structure through the temporal ordering of observed behavior. Future states define achieved goals for learning from offline experience (Andrychowicz et al., 2017; Ghosh et al., 2021), while temporal relations between states have been used to construct goal-conditioned representations and value structure (Eysenbach et al., 2022; Park et al., 2023; 2024; Wang et al., 2023). These relations encode both the remaining cost to a goal and the progress associated with an action. Our key idea is to learn this goal-relative evaluation on frozen predictive visual features and use it to shape the action preferences of a direct policy. Temporal geometry becomes a source of supervision for goal-directed action learning.

The central challenge is to connect long-horizon reachability with local action evaluation. Prior work has shown that goal-reaching relations admit structured representations that preserve temporal or directional information (Park et al., 2024; Wang et al., 2023). Multistep quasimetric learning further extends such relations across longer horizons (Zheng et al., 2026). These results suggest that temporal structure can describe not only which states are reachable, but also the remaining cost of reaching a goal from a given state.

We introduce Directed Temporal Representations for Control (DTRC) to learn this goal-relative structure from offline visual trajectories. Starting from frozen LeWorldModel (LeWM) features (Maes et al., 2026), we learn a control representation together with a directed temporal critic. Observed trajectory separation anchors the learned distance to short-range reaching cost. Bootstrapped

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.

estimates extend the same structure to longer state–goal relations. The critic combines a symmetric geometric component with an order-sensitive residual to distinguish opposite reaching directions (Liu et al., 2023). The resulting temporal cost assigns each transition a notion of goal-relative progress.

The same temporal cost also provides a common basis for policy learning. Advantage-weighted methods use value estimates to control the influence of recorded actions on policy updates (Peng et al., 2019; Kostrikov et al., 2022). DTRC applies this principle through goal-reaching temporal cost. Recorded actions receive greater weight when their transitions make greater temporal progress. Policy-generated actions can be evaluated by the same criterion through their predicted successor states.

Model-assisted learning from offline data must also control the reliability of predictions beyond the observed behavior distribution (Yu et al., 2020; 2021; Sikchi et al., 2022). DTRC restricts additional policy targets using behavior support and dynamics agreement. A policy-generated action contributes to training only when its predicted temporal cost improves upon the recorded continuation. The temporal critic places recorded behavior and model-predicted alternatives under the same goal-reaching evaluation.

We evaluate DTRC on ten visual goal-conditioned control tasks spanning navigation, reaching, and manipulation. Our experiments examine whether the learned temporal geometry acquires the structure required for control. We first study whether temporal distance preserves reaching relations beyond the local range used for direct supervision. We then test whether the learned cost distinguishes transition direction and resolves progress at the level of individual actions. Finally, we evaluate whether these properties improve goal-conditioned policy learning. Across these analyses, predictive visual features acquire a temporal organization that connects long-horizon reachability with local control decisions. Our main contributions are threefold:

- We introduce DTRC, a framework for learning directed temporal control representations on frozen predictive visual features. The learned geometry expresses state–goal relations through reaching cost and transition direction.
- We connect long-horizon temporal reachability with local action evaluation. Observed and bootstrapped trajectory relations shape the temporal geometry, while goal-relative progress provides the criterion used for policy supervision.
- Across ten visual goal-conditioned control tasks, DTRC improves goal-reaching performance and learns temporal distances that preserve reaching structure beyond the directly supervised local range. Component comparisons further show that temporal representation learning and critic-guided policy supervision improve control under the same policy parameterization.

2 PRELIMINARIES

Offline goal-conditioned control. We consider finite-horizon goal-conditioned control from visual observations. At time t, the agent observes $o_t \in \mathcal{O}$, aims to reach a goal observation $o_g \in \mathcal{O}$ within a remaining horizon $h_t \in \{1, \dots, H\}$, and selects an action $a_t \in \mathcal{A}$. The learning setting provides a fixed offline dataset $\mathcal{D}$ of unlabeled raw-pixel trajectories, where each trajectory is given by $\tau = (o_0, a_0, o_1, a_1, \dots, o_T)$. The goal-conditioned control problem is to learn a policy

$$\pi(a_t \mid o_t, o_g, h_t) \tag{1}$$

that maps the current observation, goal observation, and remaining horizon to actions that progressively steer the system toward the goal. At test time, control is closed-loop: after executing a_t, the agent receives the next observation and recomputes its action conditioned on the updated state and the same goal.

Temporal representations from visual observations. To recover control-relevant structure from high-dimensional observations, we first encode the current and goal images using a pretrained LeWM encoder E (Maes et al., 2026),

$$z_t = E(o_t), \qquad z_g = E(o_g), \tag{2}$$

where $z_t, z_g \in \mathbb{R}^{d_z}$ are compact latent representations learned from raw pixels. LeWM is a joint-embedding predictive world model whose latent space supports action-conditioned prediction and planning without reconstructing observations in pixel space. We keep the pretrained encoder fixed and use its latent features as the visual state representation for downstream control.

Although these features retain predictive information, Euclidean distance in the original LeWM latent space is not explicitly calibrated to the number of environment transitions required to reach a goal. Motivated by the Hilbert-representation view of temporal abstraction (Park et al., 2024), we use a symmetric Euclidean geometry as the metric backbone of our temporal representation. Given a LeWM latent $z \in \mathbb{R}^{d_z}$, a shared representation head $\psi : \mathbb{R}^{d_z} \to \mathbb{R}^{d_e}$ produces the learned control representation $e = \psi(z)$. A metric head $S : \mathbb{R}^{d_e} \to \mathbb{R}^{d_\phi}$ then maps this representation into a Hilbert space,

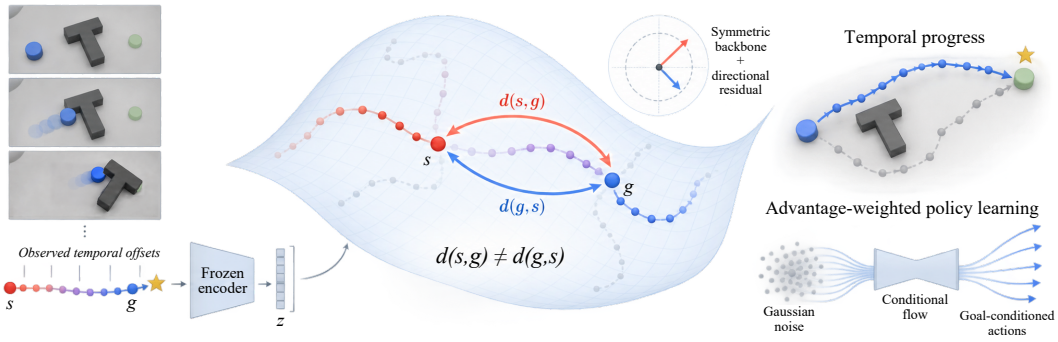
$$\phi(z) = S(\psi(z)). \tag{3}$$

We measure the symmetric temporal relation between two latent states by their Euclidean distance in this space,

$$d_{\mathrm{H}}(z_t, z_g) = \|\phi(z_t) - \phi(z_g)\|_2 . \tag{4}$$

This construction provides a symmetric geometric surrogate for temporal proximity. This construction provides a symmetric geometric surrogate for temporal proximity. However, the scalar distance remains symmetric and therefore assigns the same cost to the forward and reverse directions.

3 Directed Temporal Representations for Control



(a) Offline visual trajectories (b) Directed temporal representation (c) Policy learning

Figure 1: **Overview of directed temporal representation and policy learning. (a)** Offline visual trajectories provide observed temporal offsets as supervision. A frozen LeWM encoder maps images to visual latents. **(b)** A temporal critic built on learned control representations combines a symmetric Hilbert-style component with an order-sensitive residual, allowing forward and reverse goal-reaching costs to differ. **(c)** The critic evaluates temporal progress relative to the number of executed transitions. The resulting temporal advantage weights recorded actions for conditional policy training.

We introduce Directed Temporal Representations for Control (DTRC), an offline framework for visual goal-conditioned control. DTRC learns a temporal geometry on top of frozen predictive visual features and uses its goal-reaching costs to guide policy learning. We first define the directed temporal representation (Section 3.1). We then learn its temporal scale and long-horizon reaching structure from offline trajectories (Section 3.2). Finally, we use temporal progress to shape a direct goal-conditioned policy (Section 3.3).

3.1 Directed Temporal Representation

Goal-reaching cost has both directional and compositional structure. Opposite transition directions can incur different reaching costs, while a route through an intermediate state provides an upper bound on the direct state–goal cost (Wang et al., 2023; Liu et al., 2023). We seek a representation that preserves this path structure without imposing symmetry on temporal reachability.

We retain the Hilbert distance from Section 2 as the symmetric component and augment it with an order-sensitive residual using the metric–residual construction (Liu et al., 2023). For control representations $e_t = \psi(z_t)$ and $e_g = \psi(z_g)$, we define

$$d_\theta(e_t, e_g) = \alpha \left[\|S(e_t) - S(e_g)\|_2 + \max_{j \in [K]} [A_{\theta,j}(e_g) - A_{\theta,j}(e_t)]_+ \right], \tag{5}$$

where $[u]_+ = \max(u, 0)$, $A_\theta : \mathbb{R}^{d_e} \to \mathbb{R}^K$ defines the directed residual coordinates, and $\alpha = \exp(\eta) > 0$ sets the global scale.

The two components impose complementary structure on the learned cost. The symmetric term contributes the same component in both directions. The residual assigns additional cost through positive order-dependent coordinate differences. Reversing the state–goal order changes these differences and can produce a different reaching cost. Both components are optimized jointly by the temporal objectives introduced next.

Beyond directionality, the metric–residual construction also preserves the compositional structure of reaching cost. For any intermediate representation e_m, the direct state–goal cost is bounded by the cost of reaching e_m and continuing from e_m to the goal:

$$d_\theta(e_t, e_g) \le d_\theta(e_t, e_m) + d_\theta(e_m, e_g). \tag{6}$$

This constraint couples the costs assigned to different state–goal pairs while allowing the two reaching directions to remain distinct. The temporal objectives in the next section determine how this structured distance relates to reaching cost observed in offline trajectories.

3.2 Learning Temporal Reachability from Offline Trajectories

The directed parameterization specifies the structure of the reaching cost. Trajectory supervision supplies its temporal scale. We learn $d_\theta(e_t, e_g)$ from state–goal relations observed in offline trajectories. Following hindsight goal relabeling, future states provide achieved goals and their temporal separation supplies direct reachability supervision (Andrychowicz et al., 2017; Ghosh et al., 2021).

Long-horizon temporal targets. For a current representation e_t and a future goal e_g, let $\Delta_{t,g}$ denote their observed temporal separation when both states occur in the same trajectory. Short-range pairs provide direct measurements of reaching cost. For longer-range pairs, or state–goal pairs without a directly observed temporal separation, we compose an observed prefix with an estimate of the remaining cost:

$$y_{t,g} = \begin{cases} \Delta_{t,g}, & 0 < \Delta_{t,g} \le n, \\ n + d_{\bar\theta}(\bar e_{t+n}, \bar e_g), & \text{otherwise.} \end{cases} \tag{7}$$

Here $\bar e = \bar\psi(z)$ denotes the target representation and $d_{\bar\theta}$ denotes the corresponding target temporal distance. The first branch anchors the geometry to observed temporal separation. The second branch propagates the same scale to longer state–goal relations by combining an observed prefix with the estimated remaining cost. The bootstrap horizon n controls the length of the observed prefix.

Different trajectories and bootstrapped continuations can assign different reaching costs to similar state–goal relations. We use lower-expectile regression to emphasize shorter costs supported by the temporal targets (Newey & Powell, 1987; Kostrikov et al., 2022). The goal-conditioned temporal objective is

$$\mathcal{L}_{\text{goal}} = \mathbb{E}\left[\ell^2_{\tau_e} \left(y_{t,g} - d_\theta(e_t, e_g) \right) \right], \tag{8}$$

where $\ell^2_{\tau_e}(u) = |\tau_e - \mathbf{1}(u < 0)|\, u^2$ denotes the expectile loss and $u = y_{t,g} - d_\theta(e_t, e_g)$ is the temporal prediction residual. We choose $\tau_e < \frac{1}{2}$ to place greater regression weight on predictions that exceed their temporal targets and biases the learned distance toward lower reaching costs represented in the offline data.

Local temporal calibration. The bootstrapped objective propagates reaching costs across state–goal pairs, while nearby transitions provide direct temporal anchors. For a recorded offset $\Delta \in \{1, \dots, J\}$, we supervise $d_\theta(e_t, e_{t+\Delta})$ toward the observed separation Δ. Using the same expectile loss, we minimize

$$\mathcal{L}_{\text{local}} = \mathbb{E}_{t,\Delta}\left[\ell^2_{\tau_e} \left(\Delta - d_\theta(e_t, e_{t+\Delta}) \right) \right], \tag{9}$$

where J specifies the largest directly supervised temporal offset. These local constraints establish the short-range temporal scale used by the longer-horizon reaching objective.

Action-conditioned consistency. Temporal supervision organizes state–goal relations, while the control representation must also retain the local dynamics induced by actions. We introduce an action-conditioned transition head T_ξ and predict the next control representation with a residual update,

$$\hat{e}_{t+1} = e_t + T_\xi(e_t, a_t). \tag{10}$$

We train the transition head and control representation with

$$\mathcal{L}_{\text{cons}} = \mathbb{E}\left[\|\hat{e}_{t+1} - \bar{e}_{t+1}\|_2^2\right], \tag{11}$$

where gradients are stopped through the target representation $\bar{e}_{t+1}$. This objective keeps the learned temporal geometry compatible with the action-conditioned transitions used for control.

Overall temporal objective. We jointly train the temporal critic and control representation with

$$\mathcal{L}_{\text{critic}} = \mathcal{L}_{\text{goal}} + \lambda_{\text{local}}\mathcal{L}_{\text{local}} + \lambda_{\text{cons}}\mathcal{L}_{\text{cons}}, \tag{12}$$

where the coefficients λ_{local} and λ_{cons} control the local calibration and action-consistency terms.

3.3 Policy Learning with the Temporal Critic

We use the directed temporal critic to guide policy learning by weighting recorded actions and evaluating alternative actions proposed during training.

Advantage-weighted policy training. Flow-based policies have been used in offline reinforcement learning to represent expressive action distributions (Park et al., 2025b). We parameterize the goal-conditioned policy with conditional flow matching (Lipman et al., 2023) to represent potentially multimodal action distributions for a given current state and goal. Given a k-step action sequence $\mathbf{a}_t = (a_t, \ldots, a_{t+k-1})$, we minimize the advantage-weighted conditional flow-matching objective:

$$\mathcal{L}^{\text{data}}_{\text{policy}} = \mathbb{E}\left[w^{\text{data}}_{t,g}\,\ell_{\text{FM}}\left(\omega; \mathbf{a}_t, c_t\right)\right], \qquad w^{\text{data}}_{t,g} = \min\left\{w_{\max}, \exp\left(\frac{\mathcal{A}^{\text{data}}_{t,g}}{\beta}\right)\right\}. \tag{13}$$

where ℓ_{FM} denotes the standard conditional flow-matching loss. $c_t = [z_t, z_g, e_t, e_g, \text{Emb}(h_t/H)]$ is the policy condition and Emb denotes the horizon embedding. The temporal advantage for the recorded continuation is $\mathcal{A}^{\text{data}}_{t,g} = d_{\bar{\theta}}(\bar{e}_t, \bar{e}_g) - [k + d_{\bar{\theta}}(\bar{e}_{t+k}, \bar{e}_g)]$. Larger values assign greater weight to continuations that remain closer to an efficient temporal path. The parameter β controls the selectivity of advantage weighting, and $w_{\max}$ caps large weights.

Total policy objective. The learned temporal critic allows us to train a goal-conditioned policy from two complementary action targets. Recorded actions are weighted by their temporal advantage through $\mathcal{L}^{\text{data}}_{\text{policy}}$, while policy-generated actions provide an additional model-assisted objective $\mathcal{L}^{\text{im}}_{\text{policy}}$ when they are supported by the offline behavior and predicted to improve upon the recorded continuation. The total policy objective is

$$\mathcal{L}_{\text{policy}} = \mathcal{L}^{\text{data}}_{\text{policy}} + \lambda_{\text{im}}\mathcal{L}^{\text{im}}_{\text{policy}}, \tag{14}$$

where λ_{im} controls the contribution of the model-assisted targets. We provide the model-assisted construction and filtering procedure in Appendix B.

4 Experiments

4.1 Experimental Setup

Environments and datasets. We evaluate DTRC on ten visual goal-conditioned control tasks spanning navigation, reaching, and manipulation. We use the four offline datasets adopted by

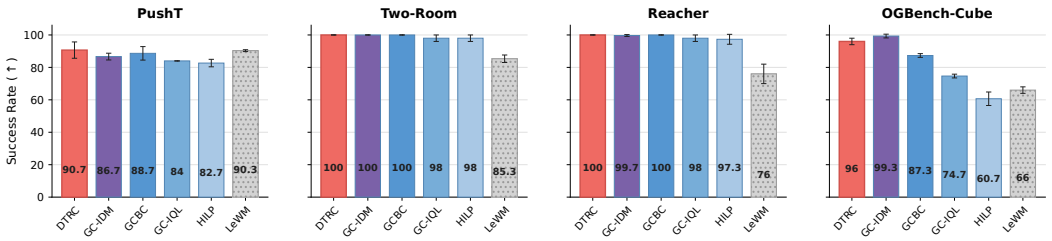


Figure 2: **Goal-conditioned control performance across the four LeWM tasks.** Results are shown for PushT (left), Two-Room (center-left), Reacher (center-right), and OGBench-Cube (right). DTRC improves substantially over LeWM on Reacher and OGBench-Cube, while matching the strongest baselines on Two-Room and remaining competitive on PushT. The smaller differences on Two-Room reflect the near-saturated performance of several direct-policy methods on this task. Bars show mean success rate over 5 evaluation seeds, with error bars denoting standard deviation.

Table 1: **Goal-conditioned control performance on six OGBench tasks.** All methods are evaluated on the same 50 start–goal pairs per task and evaluation seed with a goal offset of 25 transitions and a 50-step interaction budget. Results report mean success rate (%) $\pm$ sample standard deviation over five evaluation seeds. Best results are shown in bold and second-best results are underlined.

Method	PointMaze	Cube-Double	Scene	AntMaze-Medium	HumanoidMaze-Medium	Puzzle-3x3
LeWM	13.3 ± 1.2	50.7 ± 2.3	56.7 ± 6.4	32.0 ± 6.0	82.0 ± 2.0	24.7 ± 1.2
GCBC	28.7 ± 3.1	44.7 ± 2.3	48.7 ± 5.0	28.0 ± 3.5	83.3 ± 3.1	35.3 ± 3.1
GC-IQL	16.0 ± 2.0	48.0 ± 5.3	51.3 ± 9.0	30.0 ± 5.3	84.0 ± 2.0	31.3 ± 5.0
HILP	22.7 ± 4.6	43.3 ± 5.8	50.0 ± 7.2	28.7 ± 3.1	81.3 ± 4.2	40.0 ± 6.9
GC-IDM	$\underline{28.0 \pm 5.3}$	38.7 ± 4.6	48.7 ± 10.3	30.0 ± 3.5	88.0 ± 2.0	33.3 ± 2.3
DTRC (ours)	11.3 ± 3.1	$\underline{54.7 \pm 8.3}$	$\underline{53.3 \pm 4.2}$	$\underline{31.3 \pm 4.6}$	$\underline{84.7 \pm 5.0}$	40.0 ± 5.3

LeWM (Maes et al., 2026): PushT, TwoRoom, Reacher, and OGBench-Cube. We additionally evaluate on six visual OGBench tasks: PointMaze, Cube-Double, Scene, AntMaze-Medium, HumanoidMaze-Medium, and Puzzle-3x3. All methods receive current and goal images encoded by pretrained LeWM encoders.

Baselines. We compare against LeWM (Maes et al., 2026), goal-conditioned inverse dynamics model (GC-IDM) (Nguyen et al., 2026), goal-conditioned behavioral cloning (GCBC) (Ghosh et al., 2021), goal-conditioned IQL (GC-IQL) (Kostrikov et al., 2022; Park et al., 2025a), and a goal-conditioned adaptation of Hilbert foundation policies (HILP) (Park et al., 2024). Baseline implementations are detailed in Appendix A.

4.2 Goal-Conditioned Control Performance

We report goal-conditioned control performance in Figure 2 and Table 1. DTRC improves over LeWM by 24 and 30 percentage points on Reacher and OGBench-Cube, respectively, while remaining competitive with direct-policy baselines. It also ranks among the top two methods by mean success on five of the six additional OGBench tasks. Notably, these results are obtained with frozen visual encoders and without test-time trajectory search. They suggest that the control potential of predictive representations extends beyond latent planning: learning a goal-relative temporal structure provides an effective way to turn existing visual features into directly executable behavior.

The ablation results further demonstrate the contribution of temporal structure to policy learning (Appendix E.1). Adding the temporal representation and advantage weighting improves the same flow policy across all four LeWM tasks. This indicates that the benefit is not solely attributable to a more expressive action distribution. Temporal supervision uses recorded trajectories both as action examples and as evidence for prioritizing continuations. The learned geometry thus serves an operational role: it converts estimates of remaining goal-reaching effort into policy-training preferences, allowing reachability information to influence behavior without being queried through online trajectory optimization.

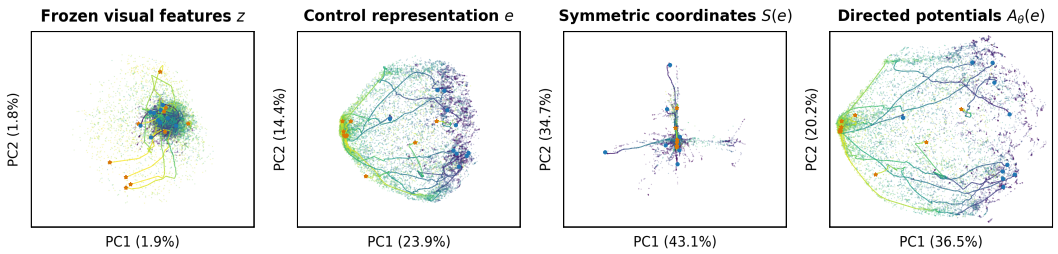


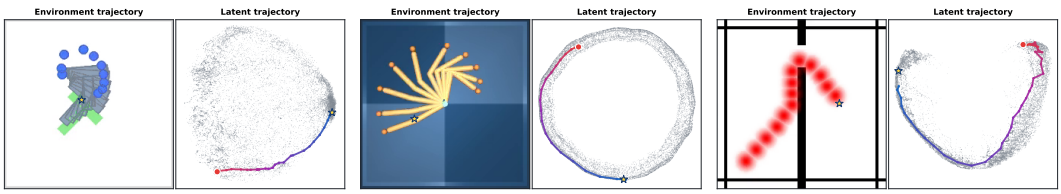
Figure 3: **Visual and temporal representations on PushT.** From left to right: frozen visual features z, control representations e, symmetric coordinates $S(e)$, and directed potentials $A_\theta(e)$. Each space is projected onto its first two principal components using the same 12,000 recorded states. Colors indicate normalized time within each episode. Curves trace recorded trajectories; circles mark their initial states and stars mark their endpoints, shown as hindsight goals. PCA is fitted independently in each space, so orientations and scales are not directly comparable across panels.

4.3 Temporal Structure of the Learned Representation

Figure 3 shows how DTRC reorganizes predictive features for temporal evaluation. Although the visual encoder remains frozen, early and late states become more visibly ordered in the learned control representation and directed-potential projection on PushT. This illustrates that control-oriented temporal structure can be learned on top of existing predictive features without modifying the visual encoder. The symmetric coordinates and directed potentials exhibit different projected organizations. These spaces are jointly trained through the temporal distance, rather than independently supervised to reproduce the same geometry. Their relevance to control therefore depends on the resulting state–goal costs, which we examine directly below.

The distance predictions preserve temporal ordering beyond the directly supervised local offsets. Across ten tasks, their rank correlations with recorded goal offsets range from 0.79 to 0.94 (Table 2). On PushT, the calibration curve increases with goal offset but increasingly falls below the recorded duration (Figure 5, left). This distinction matters because an observed offset describes the length of a particular recorded path, whereas the critic assigns a cost to the state–goal relation. The lower predictions are consistent with the lower-expectile objective's preference for shorter reaching costs, although they do not establish that those costs are achievable. The results show strong temporal ordering with less faithful absolute calibration at longer offsets.

The forward–reverse comparison further shows that the directed component contributes to the learned cost itself. On PushT, reversing a state–goal pair often produces a substantially larger prediction (Figure 5, middle). Because the symmetric term is unchanged by this reversal, the observed differences must arise from the order-sensitive residual. The temporal organization visible in the potential coordinates therefore accompanies a measurable directional effect in the critic's output. This forward preference is not uniform across tasks: the large positive mean gaps on PushT and AntMaze contrast with near-zero mean gaps on TwoRoom and Reacher (Table 2). The learned evaluation thus distinguishes ordered state pairs without requiring the same forward bias in every environment.



(a) PushT (b) Reacher (c) TwoRoom

Figure 4: **Successful closed-loop trajectories in environment and latent space.** Each example shows the same offset-100 trajectory in environment space (left) and in the learned control representation (right). Latent states are projected onto the first two principal components of a task-level PCA fitted to the corresponding DTRC representations, with gray points showing the reference latent distribution. The colored curve contains every executed decision step, and the star marks the goal.

Table 2: **Temporal structure learned by DTRC across ten tasks.** Predicted distances remain strongly correlated with recorded temporal separation, and one-step transitions usually reduce the estimated remaining cost toward distant goals. The learned asymmetry is task dependent, indicating that the directed component captures ordered reachability without imposing a uniform forward–reverse gap. ρ denotes rank correlation with recorded future temporal separation; MAE is reported in environment steps at offsets 1, 4, 8, 25, and 50; Prog. measures one-step progress toward goals 25 steps ahead; and $\Delta_{\text{rev}} = d(g, s) - d(s, g)$ measures forward–reverse asymmetry. The final three columns report latent-dynamics prediction error, disagreement–error correlation, and behavior-support AUC.

	Temporal Geometry								Model Reliability		
Environment	$\rho \uparrow$	Short MAE $\downarrow$			Long MAE $\downarrow$		Prog. $\uparrow$	Δ_{rev}	Dyn. MSE $\downarrow$	Err. Corr. $\uparrow$	Sup. AUC $\uparrow$
		1	4	8	25	50					
PushT	0.94	0.72	1.09	1.56	7.41	18.39	83.0	85.29	0.0047	0.570	0.95
TwoRoom	0.88	0.93	0.94	2.99	15.70	34.60	63.5	0.06	0.0031	0.332	0.87
Reacher	0.79	0.81	1.20	3.07	15.59	36.73	60.1	-0.01	0.0753	0.110	0.82
Cube	0.91	0.37	0.60	1.24	9.74	28.16	79.2	0.35	0.0034	0.175	0.98
PointMaze	0.89	0.40	0.62	1.31	9.09	22.50	88.0	0.79	0.0029	0.469	0.87
Cube-Double	0.90	0.67	1.21	1.78	6.05	16.74	82.5	3.16	0.0017	0.374	0.99
Scene	0.90	0.54	1.14	1.81	7.26	20.26	80.9	4.70	0.0017	0.229	0.98
AntMaze	0.84	0.72	0.95	1.79	9.00	21.09	77.6	27.76	0.0195	0.306	0.99
HumanoidMaze	0.80	1.38	1.30	3.10	12.55	28.48	62.2	0.42	0.0356	0.185	1.00
Puzzle-3x3	0.92	0.37	0.58	1.27	10.75	28.12	73.6	1.94	0.0059	0.266	0.97

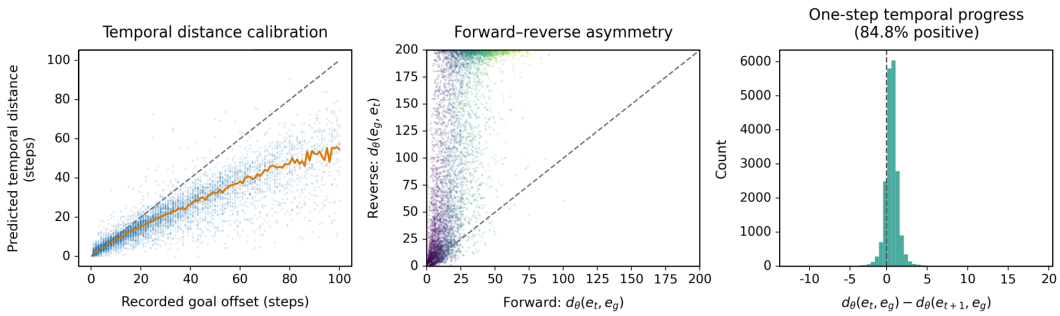


Figure 5: **Temporal-distance diagnostics on PushT.** We evaluate 20,000 recorded state–future-goal pairs with offsets of 1–100 transitions; scatter plots show 6,000 pairs. **Left:** Predicted temporal distance versus recorded goal offset. The orange curve shows the median prediction at each offset, and the dashed line marks equality with the recorded offset. **Middle:** Forward and reverse distances for the same state pair. Colors indicate recorded goal offset, from short (purple) to long (yellow). The dashed diagonal denotes equal forward and reverse costs. **Right:** Distribution of the distance reduction $d_\theta(e_t, e_g) - d_\theta(e_{t+1}, e_g)$ after one recorded transition with the goal held fixed. The vertical dashed line marks zero. Positive values indicate decreasing predicted remaining cost and account for 84.8% of the sampled pairs. Distances are capped at 200 steps.

For policy learning, temporal ordering and directionality must also be resolved at the scale of individual transitions. With the goal held fixed, the PushT progress distribution shows that one recorded transition usually decreases the predicted remaining cost (Figure 5, right). Across all ten tasks, this occurs for 60.1–88.0% of sampled pairs with goals 25 recorded transitions ahead, despite direct local supervision being restricted to $J = 8$. This provides evidence that local temporal learning supports progress evaluation toward more distant goals, rather than only fitting nearby state pairs. The learned geometry therefore connects two scales of control: state–goal distances describe longer-term reaching costs, while their changes across observed transitions provide the local signal used to construct temporal advantages.

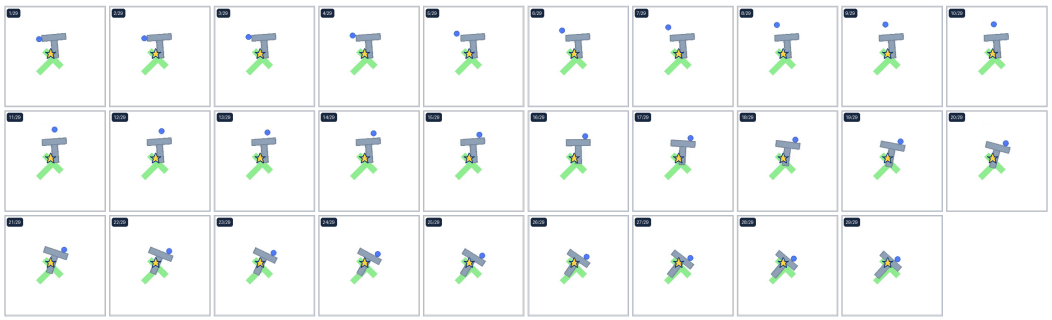


Figure 6: **Long-horizon closed-loop control on PushT.** Every environment step of a successful rollout is shown from left to right and top to bottom. The initial and goal observations are separated by 100 steps in the offline trajectory, while the learned policy reaches the target in 29 interactions within a budget of 50.

Table 3: **Trajectory quality and online computation across ten tasks.** Jerk measures action smoothness, and monotonicity is the fraction of episodes whose frozen-LeWM distance to the goal decreases at every decision step. Latency reports online action-selection time.

	Jerk ↓		Monotonicity (%) ↑		Latency (ms) ↓		
Task	DTRC	LeWM	DTRC	LeWM	DTRC	LeWM	Speedup
PushT	**0.202**	0.420	**42.0**	14.7	**38.5**	775.8	20.2×
TwoRoom	**1.027**	1.621	**25.3**	13.3	**61.7**	776.0	12.6×
Reacher	**0.614**	1.062	**64.7**	4.7	**38.8**	775.4	20.0×
Cube	**0.556**	1.544	**73.3**	44.0	**26.6**	776.6	29.2×
PointMaze	**0.351**	1.406	3.3	**4.0**	**14.1**	990.3	70.0×
Cube-Double	**1.124**	1.804	**51.3**	44.7	**13.3**	770.3	57.8×
Scene	**1.380**	1.965	62.7	**63.3**	**12.6**	780.1	61.7×
AntMaze-Medium	**2.010**	3.143	**14.7**	**14.7**	**15.4**	751.2	48.7×
HumanoidMaze-Medium	**4.397**	4.835	**75.3**	73.3	**16.3**	746.5	45.9×
Puzzle-3x3	**0.907**	1.784	84.0	**99.3**	**15.6**	763.9	48.8×

4.4 Closed-Loop Behavior and Computational Efficiency

Figures 4 and 6 show that local goal-conditioned decisions compose into coordinated long-horizon behavior. Successful rollouts need not reduce the goal distance at every step, indicating that effective control can require locally non-monotonic transitions while preserving long-horizon temporal progress. This distinction is also reflected in Table 3, where DTRC produces smooth trajectories across all ten tasks despite substantial variation in the monotonicity of the original LeWM distance.

DTRC also moves model-based evaluation from execution to training. Latent rollouts and candidate evaluation shape policy targets offline, while the learned policy performs direct closed-loop action prediction at test time. This amortizes temporal reasoning into the policy and substantially reduces online decision cost.

5 Conclusion

In this work, we show that temporal reachability provides an effective organization of predictive visual representations for goal-directed control. DTRC learns a directed temporal geometry in which latent distance reflects goal-reaching cost and temporal progress provides a criterion for action selection. Using this geometry to guide policy learning improves goal-reaching performance across diverse visual control tasks. Our findings show that predictive representations can support control more effectively when their geometry reflects the temporal structure of interaction.

AI USE STATEMENT

In this work, we used ChatGPT to improve the readability and clarity of the manuscript and to assist with organizing the structure of the paper. We did not use generative AI tools for any tasks requiring mandatory disclosure under the ICLR 2027 AI Policy for Authors. All AI-assisted edits were reviewed and, where necessary, revised by the authors. We take responsibility for the final content of this work, including text produced or modified with the aid of generative AI.

REFERENCES

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*, volume 30, 2017.

Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E. Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

Benjamin Eysenbach, Tianjun Zhang, Sergey Levine, and Ruslan Salakhutdinov. Contrastive learning as goal-conditioned reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 35, 2022. URL https://proceedings.neurips.cc/paper_files/paper/2022/hash/e7663e974c4ee7a2b475a4775201ce1f-Abstract-Conference.html.

Dibya Ghosh, Abhishek Gupta, Ashwin Reddy, Justin Fu, Coline Devin, Benjamin Eysenbach, and Sergey Levine. Learning to reach goals via iterated supervised learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=rALA0Xo6yNJ.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pp. 2555–2565. PMLR, 2019. URL https://proceedings.mlr.press/v97/hafner19a.html.

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

Nick Hansen, Hao Su, and Xiaolong Wang. TD-MPC2: Scalable, robust world models for continuous control. In *International Conference on Learning Representations*, 2024.

Nicklas Hansen, Xiaolong Wang, and Hao Su. Temporal difference learning for model predictive control. In *International Conference on Machine Learning*, 2022.

Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (gelus). *arXiv preprint arXiv:1606.08415*, 2016.

Ilya Kostrikov, Ashvin Nair, and Sergey Levine. Offline reinforcement learning with implicit q-learning. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=68n2s9ZJWF8.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *International Conference on Learning Representations*, 2023.

Bo Liu, Yihao Feng, Qiang Liu, and Peter Stone. Metric residual networks for sample efficient goal-conditioned reinforcement learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 37, pp. 8799–8806, 2023. doi: 10.1609/aaai.v37i7.26058.

Lucas Maes, Quentin Le Lidec, Damien Scieur, Yann LeCun, and Randall Balestriero. Leworld-model: Stable end-to-end joint-embedding predictive architecture from pixels, 2026. URL https://arxiv.org/abs/2603.19312.

Whitney K. Newey and James L. Powell. Asymmetric least squares estimation and testing. *Econometrica*, 55(4):819–847, 1987. doi: 10.2307/1911031.

Hoang Nguyen, Xiaohao Xu, and Xiaonan Huang. Latent geometry beyond search: Amortizing planning in world models, 2026. URL https://arxiv.org/abs/2605.08732.

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, Tobias Kreiman, and Sergey Levine. Foundation policies with Hilbert representations. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pp. 39737–39761. PMLR, 2024. URL https://proceedings.mlr.press/v235/park24g.html.

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

Seohong Park, Qiyang Li, and Sergey Levine. Flow q-learning. In *International Conference on Machine Learning*, 2025b.

Xue Bin Peng, Aviral Kumar, Grace Zhang, and Sergey Levine. Advantage-weighted regression: Simple and scalable off-policy reinforcement learning. *arXiv preprint arXiv:1910.00177*, 2019.

Boris T. Polyak and Anatoli B. Juditsky. Acceleration of stochastic approximation by averaging. *SIAM Journal on Control and Optimization*, 30(4):838–855, 1992. doi: 10.1137/0330046.

Harshit Sikchi, Wenxuan Zhou, and David Held. Learning off-policy with online planning. In *Conference on Robot Learning*, 2022.

Tongzhou Wang, Antonio Torralba, Phillip Isola, and Amy Zhang. Optimal goal-reaching reinforcement learning via quasimetric learning. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pp. 36411–36430. PMLR, 2023. URL https://proceedings.mlr.press/v202/wang23al.html.

Ying Wang, Oumayma Bounou, Gaoyue Zhou, Randall Balestriero, Tim G. J. Rudner, Yann LeCun, and Mengye Ren. Temporal straightening for latent planning. In *International Conference on Machine Learning*, 2026.

Tianhe Yu, Garrett Thomas, Lantao Yu, Stefano Ermon, James Zou, Sergey Levine, Chelsea Finn, and Tengyu Ma. MOPO: Model-based offline policy optimization. In *Advances in Neural Information Processing Systems*, 2020.

Tianhe Yu, Aviral Kumar, Rafael Rafailov, Aravind Rajeswaran, Sergey Levine, and Chelsea Finn. COMBO: Conservative offline model-based policy optimization. In *Advances in Neural Information Processing Systems*, 2021.

Bill Zheng, Vivek Myers, Benjamin Eysenbach, and Sergey Levine. Scaling goal-conditioned reinforcement learning with multistep quasimetric distances. In *International Conference on Learning Representations*, 2026.

Gaoyue Zhou, Hengkai Pan, Yann LeCun, and Lerrel Pinto. DINO-WM: World models on pre-trained visual features enable zero-shot planning. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 79115–79135. PMLR, 2025. URL https://proceedings.mlr.press/v267/zhou25t.html.

A BASELINES

All baselines operate on LeWM features extracted from current and goal images. The visual encoders are kept fixed during downstream training and evaluation. Below, we describe the control models and the adaptations used in our comparison.

A.1 LEWM

We use the pretrained LeWM encoder and action-conditioned predictor with CEM-based model-predictive control (Maes et al., 2026). Candidate action sequences are evaluated by rolling out the predictor in latent space and comparing the predicted outcomes with the encoded goal. Control proceeds in a receding-horizon manner using updated visual observations. We retain the original latent-planning formulation without adding the temporal critic or policy-learning components of DTRC.

A.2 GC-IDM

We use the goal-conditioned inverse-dynamics policy of GC-IDM (Nguyen et al., 2026). The policy takes the current LeWM feature, goal feature, and remaining horizon as inputs and predicts the recorded action through a supervised regression objective. We retain its MLP architecture and horizon-conditioning mechanism. For the additional OGBench tasks, the action output dimension is adjusted to the corresponding environment. At evaluation time, the policy predicts actions directly from the updated visual input without latent rollout or trajectory search.

A.3 GCBC

GCBC learns a direct goal-conditioned policy by supervised imitation (Ghosh et al., 2021). Future observations from the same trajectory are sampled as goals, and an MLP predicts the recorded action from the current feature, goal feature, and remaining horizon. Training minimizes the mean squared action-prediction error. This baseline uses neither a value function nor advantage weighting. It is distinct from the flow-matching behavioral-cloning variant used in our component comparisons.

A.4 GC-IQL

GC-IQL learns a goal-conditioned action-value function $Q(z_t, a_t, z_g)$ and value function $V(z_t, z_g)$ (Kostrikov et al., 2022; Park et al., 2025a). The action-value function is trained by Bellman regression using a target value network. The value function is trained by upper-expectile regression toward the target action-value estimates for recorded actions. Goal relabeling supplies a reward of zero at the relabeled goal and -1 otherwise, with bootstrapping terminated at the goal. We first train the value networks and then extract a direct goal-conditioned policy using the weighted regression procedure described below.

A.5 HILP

We adapt the Hilbert value parameterization of HILP (Park et al., 2024) to frozen LeWM features:

$$V(z_t, z_g) = - \left\| \phi_{\mathrm{H}}(z_t) - \phi_{\mathrm{H}}(z_g) \right\|_2 , \tag{15}$$

where ϕ_{H} is a learned embedding network. The value function is trained with goal-conditioned expectile temporal-difference learning. We then extract a direct goal-conditioned policy using the same weighted regression procedure as GC-IQL. This adaptation retains symmetric Hilbert value structure but replaces its directional skill learning and goal-directed skill selection with direct policy extraction. We denote this adaptation by HILP throughout the control results.

Policy extraction for GC-IQL and HILP. For both baselines, the policy is conditioned on the current feature, goal feature, and an embedding of the normalized remaining horizon. Recorded actions are fitted with a weighted Gaussian negative log-likelihood. The weights are computed from the target-value difference

$$\Delta V_{t,g} = V_{\mathrm{targ}}(z_{t+1}, z_g) - V_{\mathrm{targ}}(z_t, z_g), \qquad w_{t,g} = \min\left\{100, \exp(\alpha_{\mathrm{base}} \Delta V_{t,g})\right\}, \tag{16}$$

with $\alpha_{\text{base}} = 10$ for GC-IQL and $\alpha_{\text{base}} = 3$ for HILP. The weights are treated as fixed during policy optimization. This implementation uses the value difference rather than the $Q - V$ advantage of the original IQL policy-extraction rule. At evaluation time, both policies execute their predicted mean action.

Network parameterization. GCBC, GC-IQL, and HILP use MLPs with three hidden layers of width 512. Their policies receive a 64-dimensional sinusoidal embedding of the normalized remaining horizon. The HILP embedding ϕ_{H} has 64 dimensions. These policies predict a single action at each decision step and are queried again after receiving the next observation.

B Implementation Details

Goal sampling. For the temporal objective in Section 3.2, each current state is paired with a goal sampled from three sources. We use the next state as the goal with probability 0.1, sample a future state from the same trajectory with probability 0.6, and sample an observation uniformly from the offline dataset with probability 0.3. Future states are sampled geometrically with $\Pr(\Delta = m) = 0.02(0.98)^{m-1}$ for $m \geq 1$, with offsets beyond the remaining trajectory clipped to its final state.

Temporal-critic hyperparameters. We use a bootstrap horizon of $n = 8$ and a local supervision range of $J = 8$. The expectile parameter $\tau_e = 0.3$ is used for both $\mathcal{L}_{\text{goal}}$ and $\mathcal{L}_{\text{local}}$. The loss coefficients are $\lambda_{\text{local}} = 1$ and $\lambda_{\text{cons}} = 0.1$.

Target networks. We maintain slowly updated target networks for the control representation and directed temporal distance. After each critic update, each target parameter $\bar{w}$ is updated from its online counterpart w by exponential moving average (EMA)

$$\bar{w} \leftarrow 0.995\,\bar{w} + 0.005\,w, \tag{17}$$

following standard parameter averaging (Polyak & Juditsky, 1992). Bootstrap targets are computed entirely with the target representation and target temporal distance, with gradients stopped through all target quantities. Target-distance estimates and the resulting bootstrapped targets are capped at $d_{\max} = 200$ for numerical stability, while online critic predictions are left unclipped.

Numerical distance evaluation. For numerical stability, the Euclidean component of Eq. equation 5 is evaluated using the smoothed norm $\sqrt{r^2 + \epsilon^2} - \epsilon$, where $r = \|S(e_t) - S(e_g)\|_2$ and $\epsilon = 10^{-4}$, which preserves zero self-distance while smoothing the norm near zero.

Latent dynamics. To obtain the predicted successor $\hat{z}_{t+k}(\mathbf{a})$ in Eq. equation 24, we learn an ensemble of residual dynamics predictors in the frozen LeWM latent space. For the m-th ensemble member, the next latent state is defined as

$$\hat{z}^{(m)}_{t+1} = z_t + f_m(z_t, a_t), \qquad m \in \{1, \ldots, M\}, \tag{18}$$

where f_m predicts the action-conditioned change in the LeWM latent and M denotes the number of ensemble members. These predictors are trained separately from the pretrained LeWM predictor. They operate in the visual latent space z, whereas the transition head T_ξ used in the representation-consistency objective operates in the control representation space.

For multi-step training, each predictor is initialized at $\hat{z}^{(m)}_t = z_t$ and recursively applied under the recorded actions:

$$\hat{z}^{(m)}_{t+\ell} = \hat{z}^{(m)}_{t+\ell-1} + f_m\left(\hat{z}^{(m)}_{t+\ell-1}, a_{t+\ell-1}\right), \qquad \ell = 1, \ldots, H_F. \tag{19}$$

The predicted rollout is matched to the corresponding recorded LeWM latents with

$$\mathcal{L}_{\text{dyn}} = \mathbb{E}_{\mathcal{D}}\left[\frac{1}{M H_F d_z} \sum_{m=1}^{M} \sum_{\ell=1}^{H_F} \left\| \hat{z}^{(m)}_{t+\ell} - z_{t+\ell} \right\|_2^2\right], \tag{20}$$

where H_F is the dynamics training horizon and d_z is the LeWM latent dimension. For candidate evaluation, each ensemble member is rolled out for k transitions under the proposed action sequence $\mathbf{a}$. The successor used in Eq. equation 24 is the ensemble-mean prediction

$$\hat{z}_{t+k}(\mathbf{a}) = \frac{1}{M} \sum_{m=1}^{M} \hat{z}^{(m)}_{t+k}(\mathbf{a}), \tag{21}$$

which is mapped through the target representation $\bar{\psi}$ and then evaluated by the target temporal critic.

Model-assisted policy augmentation. As a training-time augmentation to the advantage-weighted policy objective, we use the policy to generate alternative action targets that remain supported by the offline behavior and can be predicted consistently by the learned dynamics. Specifically, under the same policy condition c_t, we sample N candidate action sequences $\{\mathbf{a}_t^{(i)}\}_{i=1}^N$ from the conditional flow policy. Before ranking candidates with the temporal critic, we filter them using a behavior-support error and a dynamics-disagreement score, since a low predicted temporal cost is informative only for candidates whose actions remain compatible with the offline data and whose predicted outcomes are consistent across the learned dynamics ensemble.

Behavior support. We train an auxiliary state-conditioned flow model v_ν on recorded actions by minimizing the standard conditional flow-matching loss (Lipman et al., 2023), with the current LeWM latent z_t as its condition. For the i-th candidate $\mathbf{a}_t^{(i)}$, we define the behavior-support error as

$$q_{\text{beh}}^{(i)} = \frac{1}{Rkd_a} \sum_{r=1}^{R} \left\| v_\nu \left(x_{s_r}^{(i,r)}, s_r, z_t \right) - \left(\mathbf{a}_t^{(i)} - x_0^{(r)} \right) \right\|_2^2, \tag{22}$$

where $x_0^{(r)} \sim \mathcal{N}(0, I_{kd_a})$, $s_r \sim \mathcal{U}(0.1, 0.9)$, and $x_{s_r}^{(i,r)} = (1 - s_r)x_0^{(r)} + s_r \mathbf{a}_t^{(i)}$. The parameter R denotes the number of flow residuals used to estimate the error. A candidate satisfying $q_{\text{beh}}^{(i)} \leq \delta_{\text{beh}}$ is considered sufficiently supported by the recorded behavior.

Dynamics agreement. For the same candidate $\mathbf{a}_t^{(i)}$, each dynamics model predicts a latent trajectory $\{\hat{z}_{t+\ell}^{(m,i)}\}_{\ell=1}^k$. We define the dynamics-disagreement score as

$$q_{\text{dyn}}^{(i)} = \frac{1}{kd_z} \sum_{\ell=1}^{k} \sum_{j=1}^{d_z} \text{Std}_{m=1,\ldots,M} \left([\hat{z}_{t+\ell}^{(m,i)}]_j \right), \tag{23}$$

where d_z denotes the LeWM latent dimension. A candidate satisfying $q_{\text{dyn}}^{(i)} \leq \delta_{\text{dyn}}$ is considered sufficiently consistent across the dynamics ensemble.

The thresholds δ_{beh} and δ_{dyn} are calibrated from the corresponding scores of recorded actions. Every 50 training updates, we compute the 90th percentile of each score in the current data batch and update the corresponding threshold by exponential moving average with retention coefficient 0.99.

We denote by $\mathcal{V}_t$ the candidate action sequences satisfying both $q_{\text{beh}}^{(i)} \leq \delta_{\text{beh}}$ and $q_{\text{dyn}}^{(i)} \leq \delta_{\text{dyn}}$. Among these candidates, we select the sequence with the lowest predicted temporal cost:

$$\mathbf{a}_t^\star = \arg\min_{\mathbf{a} \in \mathcal{V}_t} \left[k + d_{\bar{\theta}} \left(\bar{\psi} \left(\hat{z}_{t+k}(\mathbf{a}) \right), \bar{e}_g \right) \right]. \tag{24}$$

If $\mathcal{V}_t$ is empty, no model-assisted update is applied to that training sample.

We compare the selected candidate with the recorded continuation through the predicted improvement

$$\mathcal{A}_{t,g}^{\text{im}} = C_{\mathcal{D}} - C_\star, \tag{25}$$

where $C_{\mathcal{D}} = k + d_{\bar{\theta}}(\bar{e}_{t+k}, \bar{e}_g)$ is the predicted temporal cost of the recorded continuation, and $C_\star = k + d_{\bar{\theta}} \left(\bar{\psi}(\hat{z}_{t+k}(\mathbf{a}_t^\star)), \bar{e}_g \right)$ is the predicted temporal cost of the selected candidate. A positive $\mathcal{A}_{t,g}^{\text{im}}$ indicates that the candidate is predicted to improve upon the recorded continuation.

Accepted candidates provide additional flow-matching targets with weight

$$w_{t,g}^{\text{im}} = \min \left\{ w_{\max}, \exp \left(\frac{\mathcal{A}_{t,g}^{\text{im}}}{\beta} \right) \right\}, \tag{26}$$

and the corresponding model-assisted policy objective is

$$\mathcal{L}_{\text{policy}}^{\text{im}} = \mathbb{E} \left[\mathbf{1} \left(\mathcal{A}_{t,g}^{\text{im}} > 0 \right) w_{t,g}^{\text{im}} \, \ell_{\text{FM}} \left(\omega; \mathbf{a}_t^\star, c_t \right) \right], \tag{27}$$

where λ_{im} controls the contribution of the model-assisted targets. The candidate contributes to the policy update only when $\mathcal{A}_{t,g}^{\text{im}} > 0$.

C ALGORITHM AND TRAINING PROCEDURE

Algorithm 1 Training Directed Temporal Representations for Control

Input: offline trajectories $\mathcal{D}$, frozen visual encoder E

Initialize $\psi, d_\theta, T_\xi, \pi_\omega, \{f_m\}_{m=1}^{M}, v_\nu$ and target networks $\bar{\psi}, d_{\bar{\theta}}$
for $u = 1, \ldots, U$ **do**
  Sample $(o_t, \mathbf{a}_t, o_{t+k}) \sim \mathcal{D}$ and relabel goal o_g
  $z_t \leftarrow E(o_t)$, $z_g \leftarrow E(o_g)$; $e_t \leftarrow \psi(z_t)$, $e_g \leftarrow \psi(z_g)$
  Set $c_t \leftarrow [z_t, z_g, e_t, e_g, \mathrm{Emb}(h_t/H)]$
  Update $\{f_m\}_{m=1}^{M}$ with $\mathcal{L}_{\mathrm{dyn}}$ and update v_ν on recorded actions
  Compute $y_{t,g}$ with Eq. 7
  Compute $\mathcal{A}_{t,g}^{\mathrm{data}}$, $w_{t,g}^{\mathrm{data}}$, and $\mathcal{L}_{\mathrm{policy}}^{\mathrm{data}}$
  $\mathcal{L}_{\mathrm{policy}}^{\mathrm{im}} \leftarrow 0$
  if $u \geq U_{\mathrm{im}}$ **then**
    Sample $\{\mathbf{a}_t^{(i)}\}_{i=1}^{N} \sim \pi_\omega(\cdot \mid c_t)$
    Compute $\{q_{\mathrm{beh}}^{(i)}, q_{\mathrm{dyn}}^{(i)}\}_{i=1}^{N}$ and obtain $\mathcal{V}_t$
    if $\mathcal{V}_t \neq \varnothing$ **then**
      Select $\mathbf{a}_t^\star$ with Eq. 24
      Compute $\mathcal{A}_{t,g}^{\mathrm{im}}$ with Eq. 25
      if $\mathcal{A}_{t,g}^{\mathrm{im}} > 0$ **then**
        Compute $\mathcal{L}_{\mathrm{policy}}^{\mathrm{im}}$ with Eq. 27
        Refine eligible bootstrapped temporal targets
      end if
    end if
  end if
  Update ψ, d_θ, T_ξ by minimizing $\mathcal{L}_{\mathrm{critic}}$
  Update $\bar{\psi}, d_{\bar{\theta}}$ with Eq. 17
  Update π_ω by minimizing $\mathcal{L}_{\mathrm{policy}} = \mathcal{L}_{\mathrm{policy}}^{\mathrm{data}} + \lambda_{\mathrm{im}} \mathcal{L}_{\mathrm{policy}}^{\mathrm{im}}$
end for
Return: π_ω

D ARCHITECTURES AND HYPERPARAMETERS

Network architectures. We implement all trainable modules as multilayer perceptrons. Hidden layers use GELU activations (Hendrycks & Gimpel, 2016), with LayerNorm applied in the representation, policy, behavior, and dynamics networks (Ba et al., 2016). The symmetric head S, order-sensitive head A_θ, and transition head T_ξ use GELU without LayerNorm. The representation head additionally applies LayerNorm to its output. We use linear output layers throughout and do not use dropout. The directed-distance scale is initialized to $\alpha = 1$. Table 4 reports the corresponding input, hidden, and output dimensions.

Table 4: Trainable network architectures. Hidden widths exclude output layers. The policy condition has dimension $2d_z + 2d_e + 64 = 704$. $\mathrm{Emb}(s)$ denotes the 64-dimensional flow-time embedding.

Network	Input	Hidden widths	Output
Representation ψ	z	$512, 512$	128
Symmetric head S	e	256	64
Order-sensitive head A_θ	e	256	64
Transition head T_ξ	$[e, a]$	256	128
Dynamics member f_m	$[z, a]$	$512, 512$	192
Policy velocity v_ω	$[x_s, \mathrm{Emb}(s), c_t]$	$512, 512, 512$	kd_a
Behavior velocity v_ν	$[x_s, \mathrm{Emb}(s), z_t]$	$512, 512, 512$	kd_a

T_ξ predicts a residual update to e, and each f_m predicts a residual update to z. Dynamics ensemble members have independent parameters and are trained on the same minibatches.

Hyperparameters. Table 5 summarizes the default configuration.

Table 5: Default training and inference hyperparameters for DTRC.

Hyperparameter	Value
Optimization	
Training iterations U	200,000
Minibatch size	1024
Optimizer	Adam
Learning rate	3×10^{-4} (constant)
Adam moment coefficients	$(0.9, 0.999)$
Adam numerical epsilon	10^{-8}
Weight decay	0
Gradient clipping	None
Temporal critic and goal sampling	
Next / future / dataset-wide goal probabilities	$0.1/0.6/0.3$
Future-goal continuation γ_g	0.98
Bootstrap horizon n	8
Local supervision range J	8
Expectile τ_e	0.3
Local coefficient λ_{local}	1
Consistency coefficient λ_{cons}	0.1
Target-network update rate	0.005
Target distance / bootstrap cap $d_{\max}$	200
Norm-smoothing constant ϵ	10^{-4}
Flow policy	
Action horizon k	1
Horizon-conditioning cap H	50
Flow sampling solver	Forward Euler
Flow sampling steps	8
Advantage temperature β	1
Maximum weight $w_{\max}$	20
Action clip (standardized space)	$[-3, 3]$
Test-time policy samples per decision	1
Model-assisted augmentation	
Dynamics ensemble size M	4
Dynamics training horizon H_F	4
Model-assisted start iteration U_{im}	10,000
Policy candidates per condition N	16
Behavior-score samples R	2
Model-assisted coefficient λ_{im}	1

E Additional Results

E.1 Component Ablations

Table 6: **Ablation study on the four LeWM tasks.** Success rates (%) are reported as mean ± sample standard deviation over five evaluation seeds. All variants use the same 50 start–goal pairs per task and seed, a goal offset of 25 recorded transitions, and a 50-step interaction budget.

Configuration	PushT	TwoRoom	Reacher	Cube
Flow BC	74.7 ± 4.6	94.7 ± 4.2	68.0 ± 3.5	91.3 ± 1.2
Flow BC + temporal critic	84.0 ± 5.3	100.0 ± 0.0	100.0 ± 0.0	97.3 ± 3.1
DTRC w/o behavior support	88.0 ± 4.0	100.0 ± 0.0	100.0 ± 0.0	95.3 ± 1.2
DTRC w/o directional residual	90.0 ± 5.3	100.0 ± 0.0	100.0 ± 0.0	$\mathbf{98.7 \pm 2.3}$
DTRC (full)	$\mathbf{90.7 \pm 5.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	96.0 ± 2.0

Configuration details. Flow BC uses unweighted conditional flow matching. Flow BC + temporal critic adds the learned control representation and temporal-advantage weighting, without model-assisted augmentation. The behavior-support ablation retains the dynamics-agreement filter, while the directional-residual ablation retains only the symmetric distance component. The full model includes both model-assisted policy targets and bootstrap-target refinement.

E.2 Performance across Goal Offsets

Table 7: **Goal-conditioned control across goal offsets.** Task-averaged success rates (%) over six OGBench tasks.

	Goal offset		
Method	25	50	100
LeWM	43.3 ± 1.2	17.6 ± 4.1	10.7 ± 1.9
GCBC	43.8 ± 1.5	16.4 ± 4.6	9.1 ± 0.8
GC-IQL	43.9 ± 2.2	16.9 ± 3.8	9.8 ± 1.0
HILP	45.1 ± 1.3	17.2 ± 2.7	9.0 ± 0.9
GC-IDM	44.4 ± 2.5	16.8 ± 3.3	8.3 ± 0.3
DTRC (ours)	$\mathbf{45.6 \pm 4.2}$	$\mathbf{20.3 \pm 4.7}$	$\mathbf{18.9 \pm 0.7}$

Evaluation details. The six tasks are PointMaze, Cube-Double, Scene, AntMaze-Medium, HumanoidMaze-Medium, and Puzzle-3x3, with equal weight in the task average. Within each goal offset, all methods use the same 50 start–goal pairs per task and evaluation seed. We first average success rates across tasks for each seed, then report the mean and sample standard deviation over 5 evaluation seeds. Goal offsets count recorded trajectory transitions rather than minimum reaching times. HILP denotes the Hilbert-value adaptation used in our visual-control comparison. Checkpoints are fixed across goal offsets and evaluation seeds, so the reported deviations do not measure variability across independent training runs.

F REPRESENTATION COMPARISON ACROSS METHODS

Figure 7 compares representations and decision features across six methods on all ten tasks. Rows correspond to tasks. Columns show LeWM, DTRC, HILP, GCBC, GC-IQL, and GC-IDM from left to right. LeWM, DTRC, and HILP provide state representations. GCBC, GC-IQL, and GC-IDM provide goal-conditioned decision features.

For each task, all methods use the same 12,000 recorded state indices and associated future goals. Goals lie 25 recorded transitions ahead, or at the episode endpoint when fewer transitions remain. We project each method's features onto their first two principal components using a separate PCA fit. Colors indicate normalized time within each episode, from red at the beginning to blue at the end. The projections provide a qualitative view of feature organization; their orientation, scale, and absolute coordinates are not directly comparable across methods.

Checkpoint scope. For PushT, TwoRoom, Reacher, and OGBench-Cube, the HILP, GCBC, and GC-IQL panels use preserved state-vector-input checkpoints. The figure therefore compares feature organization across the available checkpoints rather than isolating method differences under identical visual inputs.

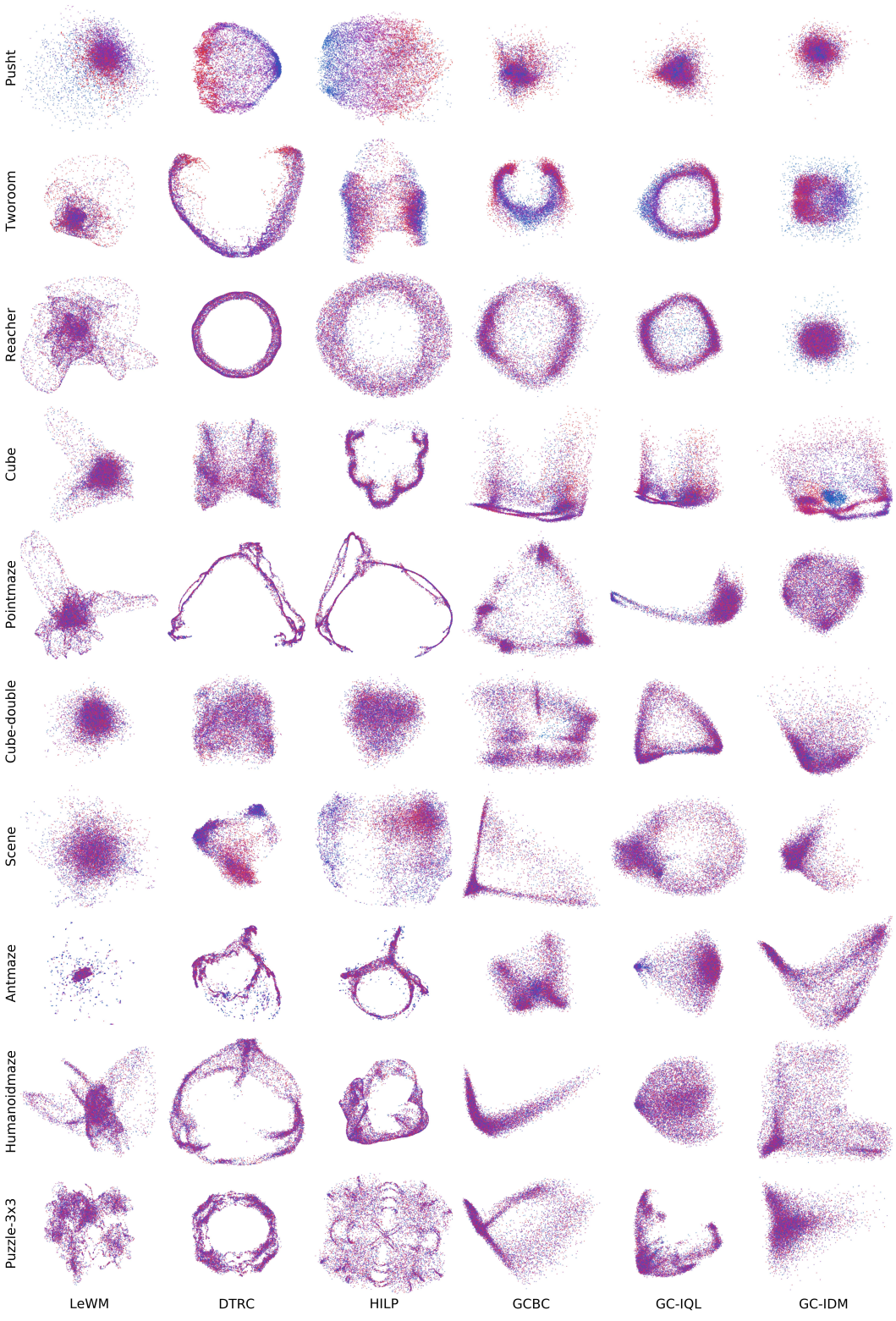


Figure 7: **Representations and decision features across six methods and ten tasks.** Colors indicate normalized episode time from red to blue.

G LATENT-SPACE VISUALIZATIONS

We visualize the representations of DTRC across all ten tasks. Each figure shows four spaces from left to right: frozen visual features z, the control representation e, symmetric coordinates $S(e)$, and directed potentials $A_\theta(e)$. For each task, we use the same 12,000 recorded states in all four panels and fit principal component analysis (PCA) separately in each space. The axis labels report the explained variance of the first two principal components. The learned representations and distance heads come from the target networks; the figure labels omit target-network bars. PCA coordinates and scales are not directly comparable across panels, and distances in these projections do not represent the learned temporal distance.

Colors indicate normalized time within each recorded episode, from its beginning in purple to its end in yellow. The overlaid curves connect temporally ordered states from ten recorded trajectories, with up to 100 states per trajectory. Blue circles mark initial states. Orange stars mark trajectory endpoints, which serve as hindsight goals in the visualization. These curves show offline trajectories rather than policy rollouts.

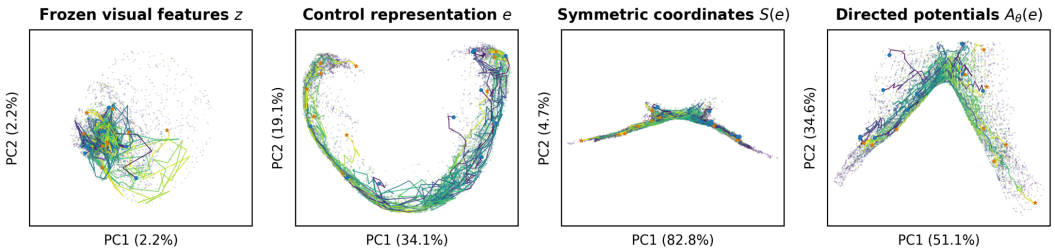


Figure 8: **Latent-space visualizations on TwoRoom.**

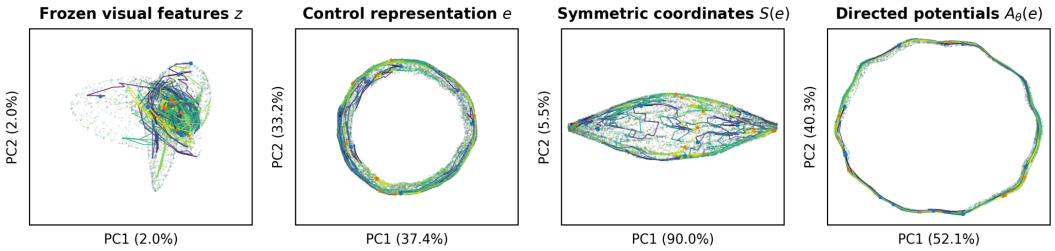


Figure 9: **Latent-space visualizations on Reacher.**

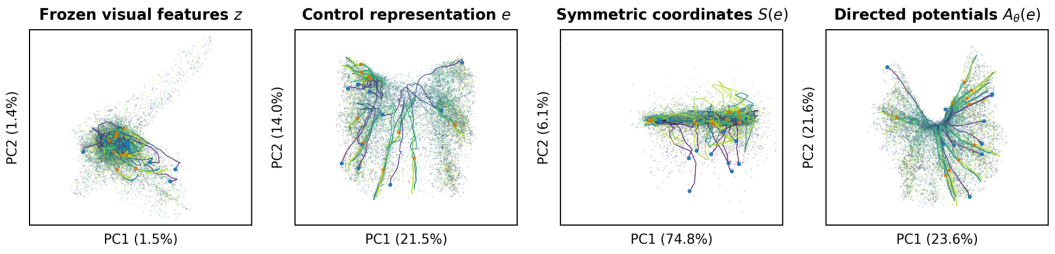


Figure 10: **Latent-space visualizations on OGBench-Cube.**

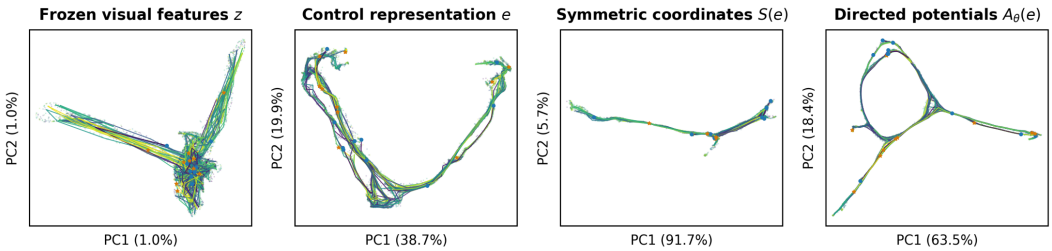


Figure 11: **Latent-space visualizations on PointMaze.**

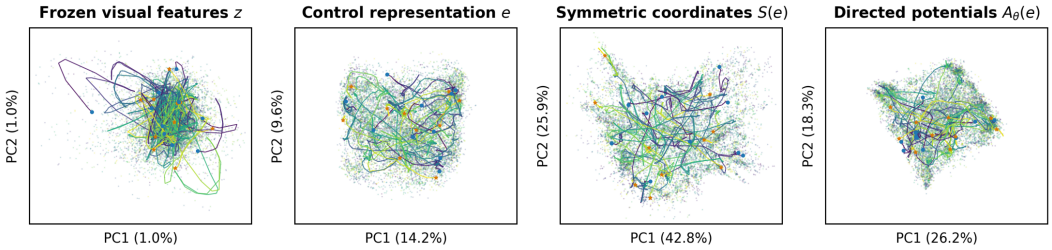


Figure 12: **Latent-space visualizations on Cube-Double.**

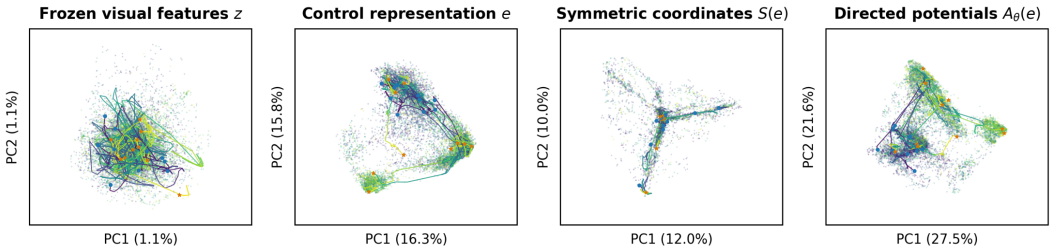


Figure 13: **Latent-space visualizations on Scene.**

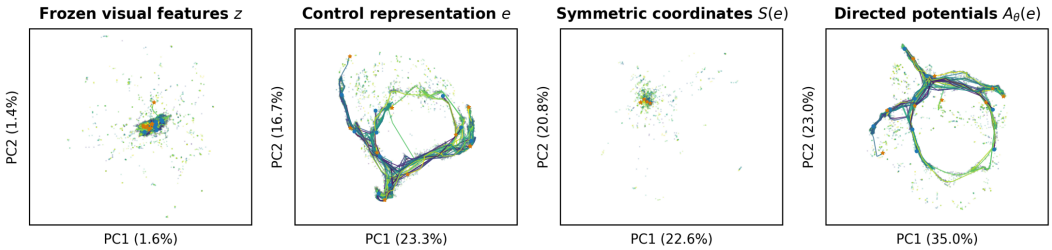


Figure 14: **Latent-space visualizations on AntMaze-Medium.**

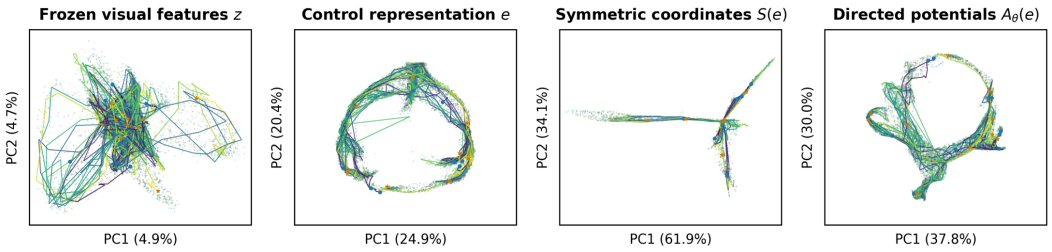


Figure 15: **Latent-space visualizations on HumanoidMaze-Medium.**

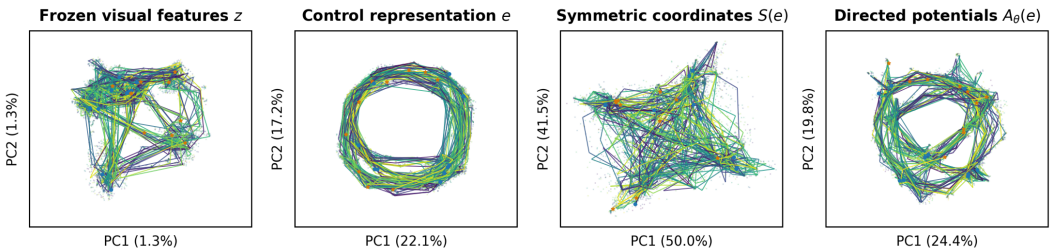


Figure 16: **Latent-space visualizations on Puzzle-3x3.**

H TEMPORAL-DISTANCE DIAGNOSTICS

We examine temporal-distance calibration, forward–reverse asymmetry, and one-step temporal progress across all ten tasks. For each task, we sample 20,000 current–future state pairs from recorded episodes. For each current state, we sample uniformly among future states within the next 100 transitions without crossing the episode boundary. The scatter plots display a subset of 6,000 pairs. The median curves and progress histograms use all sampled pairs. All distances come from the target temporal critic evaluated in the full representation space, with outputs capped at 200 environment steps. The notation below and the figure labels omit target-network bars.

Temporal distance calibration. The left panel plots predicted temporal distance against the recorded goal offset. The orange curve shows the median prediction at each offset. The dashed diagonal marks equality between predicted distance and recorded offset. This line provides a reference for observed trajectory duration, not a ground-truth minimum reaching cost.

Forward–reverse asymmetry. The middle panel compares $d_\theta(e_t, e_g)$ on the horizontal axis with $d_\theta(e_g, e_t)$ on the vertical axis for the same state pair. Colors indicate recorded goal offsets, from smaller offsets in purple to larger offsets in yellow. The dashed diagonal marks equal forward and reverse costs. Points above the diagonal indicate a higher predicted cost in the reverse direction.

One-step temporal progress. The right panel shows the distribution of $d_\theta(e_t, e_g) - d_\theta(e_{t+1}, e_g)$, where e_{t+1} corresponds to the recorded next state and the goal remains fixed. The dashed vertical line marks zero progress. Positive values indicate a decrease in predicted remaining cost. The percentage in the panel title reports the fraction of pairs with positive progress. These histograms aggregate goals across the sampled offset range, rather than only the offset-25 goals used for the progress statistic in Table 2.

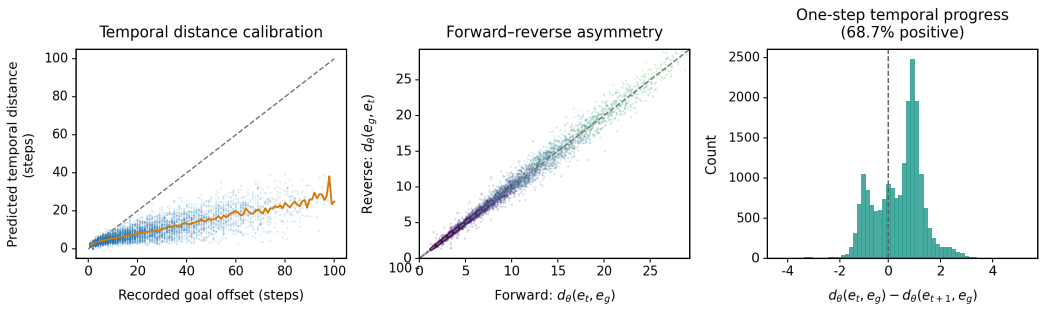


Figure 17: **Temporal-distance diagnostics on TwoRoom.**

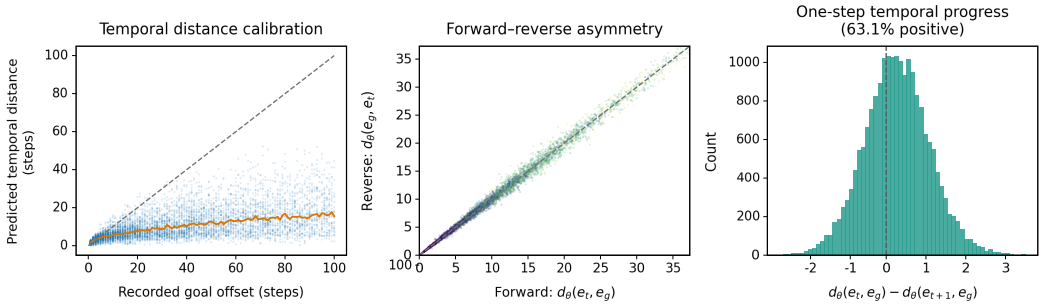


Figure 18: **Temporal-distance diagnostics on Reacher.**

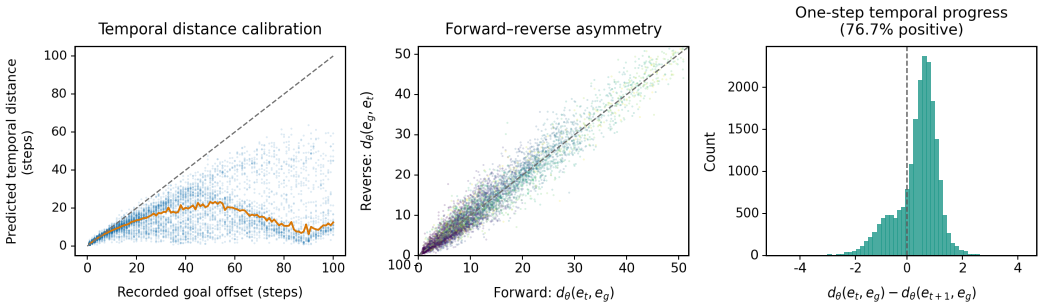


Figure 19: **Temporal-distance diagnostics on OGBench-Cube.**

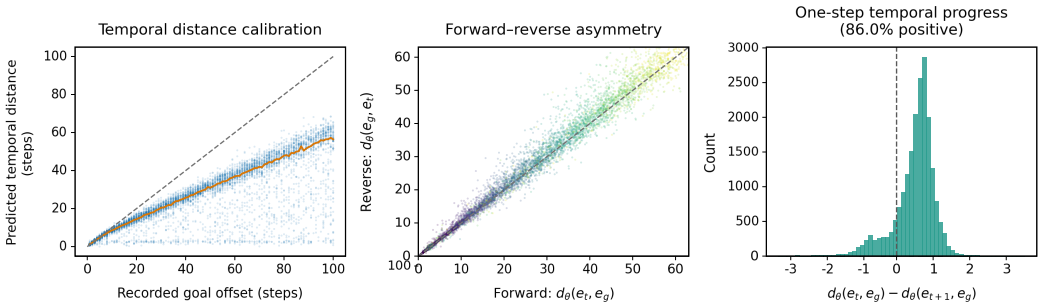


Figure 20: **Temporal-distance diagnostics on PointMaze.**

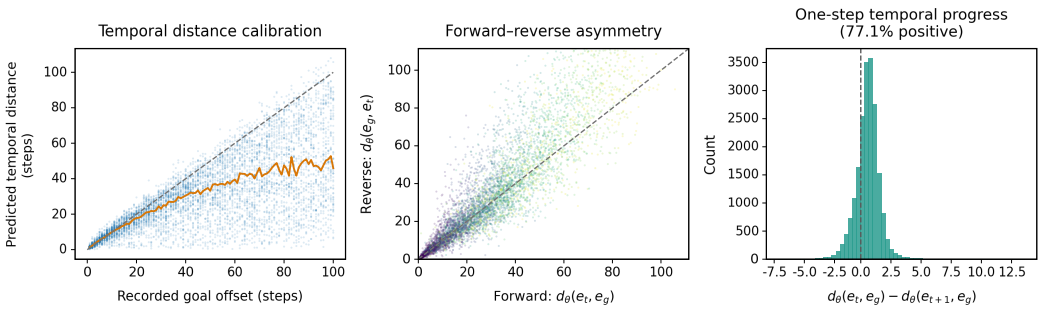


Figure 21: **Temporal-distance diagnostics on Cube-Double.**

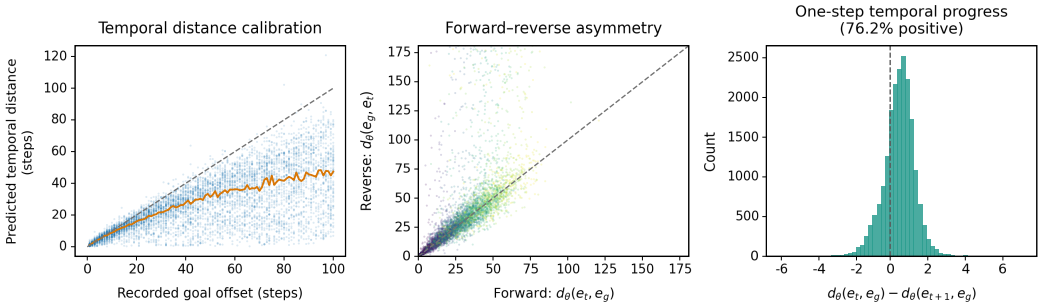


Figure 22: **Temporal-distance diagnostics on Scene.**

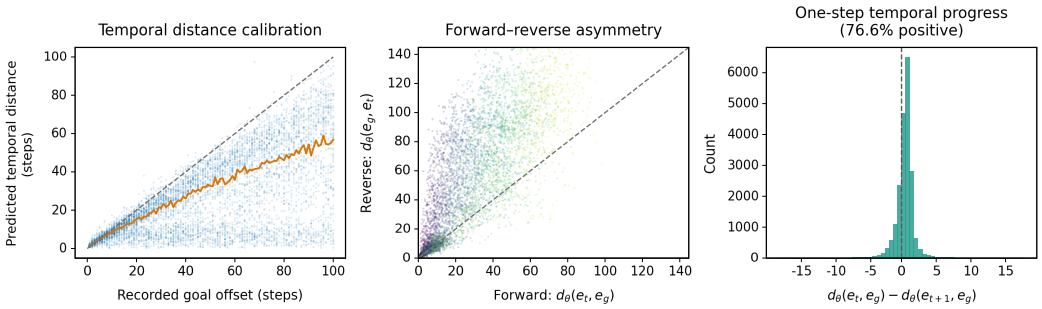


Figure 23: **Temporal-distance diagnostics on AntMaze-Medium.**

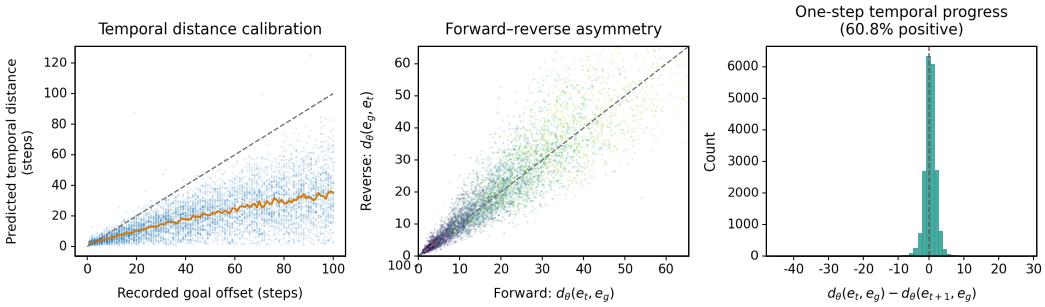


Figure 24: **Temporal-distance diagnostics on HumanoidMaze-Medium.**

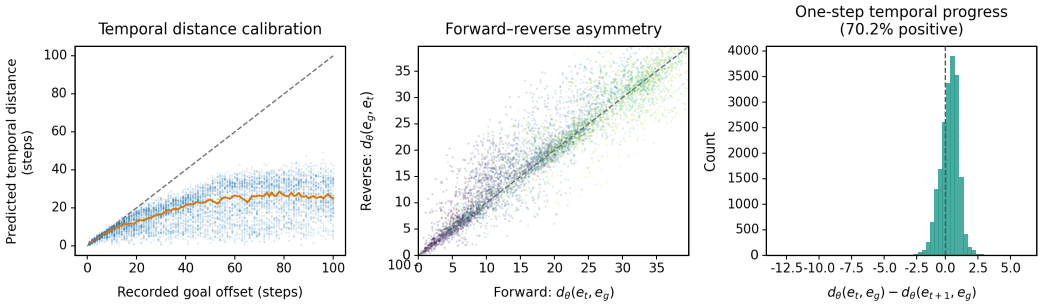


Figure 25: **Temporal-distance diagnostics on Puzzle-3x3.**