DOUBLE IDENTITY: DECOUPLING CONTROL STATE AND TASK BELIEF IN REINFORCEMENT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In partially observable control problems with persistent hidden task properties, immediate control and task inference may benefit from different temporal contexts. Conditions relevant to the next action can change at every transition, while evidence identifying an unobserved goal or dynamics regime accumulates over longer horizons. Encoding both in a single representation creates conflicting demands: the representation must be highly responsive to local changes while preserving as much task-relevant information from prior interactions as possible. We introduce State-Task Adaptive Control (STAC), an actor-critic architecture that assigns these inference roles to separate causal encoders with different temporal contexts and learning objectives. A control encoder summarizes recent interactions and is trained through temporal-difference learning. At the same time, a task encoder attends causally to the full interaction history and is trained with goal supervision, reward prediction, and contrastive objectives. A trainable adapter maps the task encoder's output to belief features used by the actor and critics. Under a hidden-goal protocol using the ten Meta-World ML10 training tasks, STAC achieves mean evaluation success rates of 71% within the training region and 62% in held-out regions, exceeding the strongest evaluated baseline on each split by 20% and 18%, respectively. Architectural ablations further support limiting the control encoder to recent interactions while using a separate encoder to aggregate available task evidence. Together, these findings show that tailoring both temporal context and learning objectives to the distinct demands of control-state estimation and task inference can improve learning and generalization in the evaluated manipulation setting.

1 INTRODUCTION

An agent operating under uncertainty about its task or environment must infer both its current physical state and the persistent properties that help determine which behaviors will lead to success. Although these inference problems must be learned using the same stream of interaction, they need not benefit from the same temporal representation. Recent observations may be sufficient to track local motion and object contact, whereas evidence about an unobserved goal or environment parameter can accumulate over longer timescales. An effective representation must therefore support two potentially conflicting inference problems: maintaining rapidly changing information needed for control while integrating evidence about quantities that remain fixed throughout a trial.

The distinction between state and task inference is well established in meta-reinforcement learning. Prior methods condition control on inferred task variables (Rakelly et al., 2019; Humplik et al., 2019), and hierarchical latent-variable models explicitly distinguish hidden state from hidden task (Akuzawa et al. 2021). However, a conceptual factorization into state and task variables does not by itself determine how those representations should be learned. In particular, it leaves open two consequential design choices: *which portions of interaction history each representation should access, and which learning signals should update each representation*. Our central question is therefore not simply whether state and task should be represented separately, but how their distinct inference roles should shape the temporal and optimization structure of the learning system.

To this end, we introduce State-Task Adaptive Control (STAC), which uses a bounded history to learn a control representation, while using the entire trial history to learn the persistent task repre-

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.

sentation. The two representations are trained using objectives aligned with their respective goals. Temporal-difference learning shapes the control representation, whereas goal supervision, reward prediction, and contrastive objectives train the task encoder. A trainable adapter then exposes the task representation to the actor and critics without allowing value-learning gradients to flow directly into the task encoder. STAC investigates how temporal access should be allocated between control and task inference. Prior work already combines recurrent control with separately supervised belief networks and blocks reinforcement learning gradients into belief inference (Humplik et al. 2019). Building on this structure, STAC explicitly bounds the control encoder's receptive field while preserving full-trial access in the task encoder, and evaluates this allocation against full-history control and shared-encoder alternatives.

Our main contributions are as follows.

- We separate state and task inference not only in what is encoded, but also in how it is learned, preventing the two inference problems from competing for a shared encoder's capacity or gradients.
- We instantiate this separation in STAC, a framework that jointly learns two representations with separate encoders, each with their own history horizon, parameters, and learning signals to simultaneously satisfy distinct, potentially conflicting demands.
- We show that this structure substantially improves both learning and generalization on Meta-World ML10, achieving approximately 71% average success on in-support configurations similar to those seen during training and 62% on held-out configurations outside the training distribution, compared with 51% and 44% for the strongest respective baselines.
- The clearest controlled ablation supports bounded control history. Restricting the control encoder's access to recent interactions increases in-support task success by 27% with unchanged parameter count and learning objectives. Shared full-history encoders perform even worse, whereas the experimental results do not establish a meaningful performance benefit from blocking TD gradients into the task encoder.

2 Related Work

History-conditioned reinforcement learning. RL^2 (Duan et al., 2016) trains recurrent policies to adapt from interaction history. TrMRL and AMAGO extend this approach using attention and long-sequence training, while AMAGO-2 addresses optimization across heterogeneous tasks (Melo, 2022; Grigsby et al., 2024a;b). These methods learn history representations for action selection without prescribing separate control and task channels with different history windows. They show that the optimal history context length is heavily task dependent and non-monotone, suggesting that a single receptive field is ill-suited for producing both a representation for control and one for task inference. Ni et al. (2021) validates this methodological family by demonstrating that recurrent agents are strong POMDP baselines. ECET (Shala et al., 2025) further illustrates the importance of cross-episode information.

Explicit task inference. The second group of methods requires learning a task-belief variable, that is paired with unprocessed observations rather than a learned control representation. HiP-MDPs (Doshi-Velez & Konidaris, 2016) is an early example of this pattern, where different tasks are modeled as a library of MDPs that are weighted by an instance-specific parameter vector, which dictate the specific dynamics and construction of a particular task. PEARL (Rakelly et al., 2019) infers a probabilistic context variable representing the underlying task by feeding unordered transitions through a permutation-invariant encoder. VariBAD (Zintgraf et al., 2020) learns an approximate task belief to support adaptation and exploration. ContraBAR (Choshen & Tamar, 2023) learns a history representation through contrastive prediction and develops information-state guarantees under assumptions on its predictive objective. In each of these methods, the policy and critic are trained using an explicit belief representation, that is appended to unprocessed environment states. STAC instead routes the observation history through a dedicated control encoder before it reaches the policy. This contrast involves the agent interface; it does not imply that prior policy networks lack learned state features, or that these methods cannot be extended with observation encoders.

Separating hidden state and hidden task. The closest precedents explicitly model this distinction. Humplik et al. (2019) train a separate belief stream using privileged task information. HiP-BMDP (Zhang et al., 2020) learns state abstractions together with environment representations.

Table 1: Comparison of representation and learning structure across approaches. *: Task representation is inferred from transition sequences. †: Control uses the current state/observation rather than a learned history-based control representation.

Method	Task repr.	Control repr.	Different temporal contexts	Task-inference objective
RL^2/TrMRL/AMAGO-2	✗	✗	✗	✗
PEARL	✓*	✗†	✓	✗
VariBAD/ContraBAR	✓*	✗†	✓	✓
Humplik et al. (2019)	✓	✓	✗	✓
HSSM	✓	✓	✗	✓
ECET	✓	✗†	✓	✗
STAC	✓	✓	✓	✓

HSSM (Akuzawa et al., 2021) introduces global task and local state latents within a learned generative model, providing an example of the same conceptual state-task factorization that we are concerned with. STAC makes these distinctions explicit by computing the control representation from recent interactions while allowing the task representation to be learnt from a longer horizon.

3 Problem Formulation

We consider a family of partially observable control problems indexed by a latent task variable Θ. At the beginning of each trial, a task is sampled as $\Theta \sim p(\theta)$ and remains fixed for T interactions. Conditional on $\Theta = \theta$, the latent physical state transitions according to:

$$x_{t+1} \sim P_\theta(\cdot|x_t, a_t), \tag{1}$$

while the agent receives observation o_t and reward r_t. The task variable represents properties that persist throughout the trial, such as the goal or environment dynamics. A trial may contain multiple episodes, where an episode reset reinitializes the physical state x_t but does not resample Θ. Assuming r_t is observed after action a_t, the history available when selecting a_t is:

$$H_t = (o_0, a_0, r_0, o_1, ..., a_{t-1}, r_{t-1}, o_t). \tag{2}$$

Given the interaction history H_t, the agent's Bayesian belief is a joint distribution over the current latent physical state and the persistent task:

$$\mathcal{B}_t(x, \theta) = p(X_t = x, \Theta = \theta|H_t). \tag{3}$$

This joint belief allows dependence between physical state and task uncertainty, as the inferred physical state can depend on the hypothesized task θ. Approximating this posterior requires two distinct inference problems. Estimating the physical state requires tracking quantities that may change rapidly within an episode (e.g., an object's velocity), whereas task inference requires combining evidence across episodes about the fixed task and goal. When the environment model is known, $\mathcal{B}_t$ is sufficient for Bayes-optimal control. However, in our assumed setting, this posterior is unavailable and a finite-dimensional embedding is not guaranteed to retain all information required for optimal control. We therefore learn two summaries of the interaction history:

$$z_t = f_\phi(H_t^{(K)}), \quad b_t = A_\eta(g_\psi(H_t)), \tag{4}$$

where $H_t^{(K)}$ represents a window containing the K most recent interactions and the current observation. The control representation z_t should retain information about the agent's current physical state, whereas the task representation b_t summarizes evidence about the task and goal. The policy and critics operate on the combined representation $s_t = [z_t||b_t]$:

$$\text{action selection: } a_t \sim \pi_w(\cdot|s_t), \quad \text{evaluation: } Q_{\nu_i}(s_t, a_t), \text{ for } i \in \{1, 2\}. \tag{5}$$

We treat b_t as a task representation rather than the exact posterior $\mathcal{B}_t$. Because the agent's learning task Θ remains fixed throughout a trial and H_{t+1} contains H_t, additional interactions cannot decrease task information available in the history, as formalized in Proposition 1 (Appendix B.1). This need not hold for the learned task representation: the data-processing inequality gives $I(\Theta; b_t) \leq I(\Theta; H_t)$ but does not imply $I(\Theta; b_{t+1}) \geq I(\Theta; b_t)$. This gap between available and retained task information is one of the fundamental challenges for learned task inference

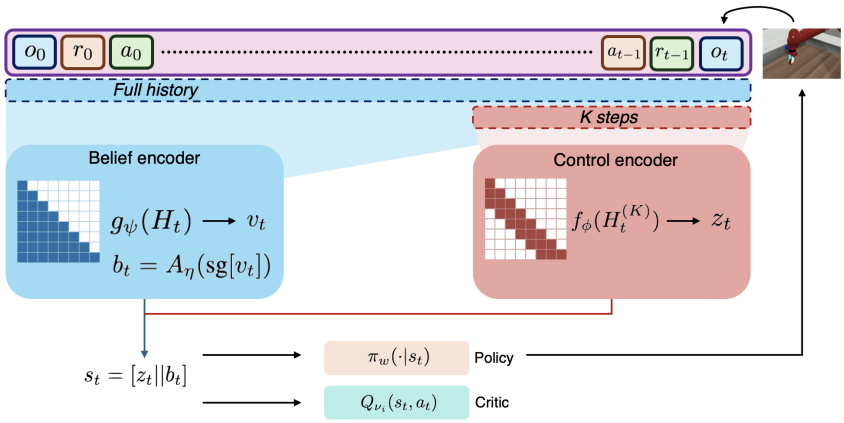


Figure 1: STAC concat architecture. The task encoder maps the full trial history H_t to a task latent b_t, while the control encoder maps a recent K-step window $H_t^{(K)}$ to a control latent z_t. Their concatenation conditions the SAC policy and critics.

and motivates STAC's full-history task inference. The policy and critic networks are trained using SAC (Haarnoja et al., 2018) with replayed trials and the discounted maximum-entropy objective:

$$J(\pi) = \mathbb{E}\left[\sum_{t=0}^{T-1} \gamma^t (r_t + \alpha \mathcal{H}[\pi(\cdot|z_t, b_t)])\right] \tag{6}$$

4 Method: State-Task Adaptive Control

Fig. 1 illustrates how STAC instantiates the two history summaries from Eq. 4 with two causal encoders that process the same interaction history but differ in what they see and in what objectives are used to train each representation. The *control encoder* f_ϕ attends only to the K most recent interactions and is trained by temporal-difference (TD) learning, so that the *control state* z_t tracks the rapidly changing conditions that help determine the next optimal action. The *task encoder* g_ψ attends to the entire trial and is trained by task-inference objectives, enabling its output to accumulate evidence about quantities that stay fixed throughout a learning trial. A small adapter is then applied to the task encoder's output to produce the *task belief* b_t used by the actor and critics.

4.1 Time-scale-specific Causal Encoders

Both encoders access the same sequence of tokens. At step t, the token

$$\xi_t = [o_t \,||\, a_{t-1} \,||\, \mathrm{symlog}(r_{t-1}) \,||\, c_t\,] \in \mathbb{R}^{48}, \quad \text{where } \mathrm{symlog}(r) = \mathrm{sgn}(r)\log(1+|r|), \tag{7}$$

pairs the current observation o_t with the previous action and reward, similar to RL^2 and AMAGO, while $c_t \in \mathbb{R}^4$ specifies the within-episode step, reset indicator, episode index, and within-trial step. Every component is available before a_t is selected, and because symlog is invertible, the token sequence $\xi_{0:t}$ carries exactly the information present in H_t. Before each trial, the previous action and reward are initialized as: $a_{-1} = 0$, $r_{-1} = 0$. Once an episode is completed, (a_{t-1}, r_{t-1}) retain the final action and reward from the preceding episode.

Bounded context control encoder. Each encoder embeds tokens with its own two-layer MLP, and then processes them using a causal transformer. The two encoders share the same transformer architecture, but have different attention masks. The control encoder applies a banded causal mask $M_W[q,k] = \mathbb{I}[0 \le q-k \le W]$, meaning a query at position q can only attend to keys at positions $q-W, ..., q$. Stacking D such layers gives the output at step t a receptive field of $K = DW + 1$ tokens.

$$z_t = f_\phi(H_t^{(K)}), \quad H_t^{(K)} = \xi_{\max(0,t-K+1):t}. \tag{8}$$

Enforcing this receptive-field requires local masking in every attention layer as a recent output from a transformer with full attention could preserve a dependence on earlier tokens. This bound ensures

that z_t is focused on the current physical situation, since previous interactions cannot interfere with estimating the present one. The bound restricts access and not content, therefore z_t is not task-independent as recent observations, rewards, and actions can all carry task information.

Full history task encoder. The task encoder uses unrestricted causal attention over the whole trial, so all task evidence in H_t, which cannot decrease as the trial proceeds (Proposition 1), remains available at every decision step. Three linear heads map the encoder's output to the task statistics:

$$v_t = g_\psi(H_t) = [u_t||\mu_t||\ell_t], \tag{9}$$

where u_t is a unit-norm task embedding, and μ_t and ℓ_t are the mean and diagonal log-variance of a Gaussian estimate of the unobserved goal. The embedding is organized by task attributes, while (μ_t, ℓ_t) report the current goal estimate together with its uncertainty. Full attention allows the encoder to access all available evidence, but does not guarantee that it is retained (Proposition 2), and thus improving retention is the role of the task encoder's inference objectives.

4.2 Combining control and task representations

Task statistics v_t are passed through a small adapter and are mapped to the belief input b_t before reaching the actor and critics:

$$b_t = A_\eta(\text{sg}[v_t]) = \text{LayerNorm}(W_2\text{GELU}(W_1 v_t)) \in \mathbb{R}^{32} \tag{10}$$

The stop-gradient $sg[\cdot]$ separates learning task information from learning to use that information for control. We detach the task encoder's outputs before passing them to the adapter, so the TD loss can update the critics, control encoder f_ϕ, and adapter without propagating into the task encoder. This design isolates task inference from direct TD updates. During actor updates, the combined representation is detached before being passed to both the actor and critics, so the policy loss updates neither encoder nor the adapter. This follows the same principle as SAC+AE (Yarats et al., 2019), which blocks actor gradients from updating the encoder.

We consider two methods to merge control and task representations for use by the actor and critics. **STAC concat** merges them at the input: the MLP actor receives $s_t = [z_t||b_t]$, while each of the two MLP critics receives $[s_t||a_t]$. **STAC gating** instead uses b_t to modify the network's final hidden features. We assume each network $N \in \{\pi_w, Q_{\nu_1}, Q_{\nu_2}\}$ receives the base input $x_N = z_t$ for the actor and $x_N = [z_t||a_t]$ for a critic, while $F_N(x_N)$ denotes its final hidden features and O_N is its output layer. The gated network computes

$$N(x_N, b_t) = O_N\left(F_N(x_N) + \kappa_N(V_N n(b_t)) \odot \tanh(C_N(x_N))\right). \tag{11}$$

Here $n(\cdot)$ is a parameter-free LayerNorm, V_N and C_N are learned linear maps, $\odot$ is elementwise multiplication and κ_N is a learned scalar. The belief determines what is added to the features, while the gate, computed from the current control context, determines how much of each component is used. κ_N is initialized to zero, thus each network initially depends on the control state alone and learns how strongly to rely on the belief as learning progresses.

4.3 Learning objectives and gradient ownership

The actor and twin critics are trained off-policy with SAC (Haarnoja et al. 2018) from a replay buffer of complete trials to increase sample efficiency. Because both encoders are learned, all representations are recomputed from the stored tokens at every update rather than stored in the buffer. Bootstrapping continues across internal episode resets and terminates only at the trial boundary. To balance learning across heterogeneous task families, value normalization and entropy temperature are maintained separately for each family. The task encoder and an auxiliary reward head R_χ are trained with the following objective, whose terms are defined in Appendix A.2:

$$\mathcal{L}_{task} = \lambda_1\mathcal{L}_{goal} + \lambda_2\mathcal{L}_{rew} + \lambda_3\mathcal{L}_{attr} + \lambda_4\mathcal{L}_{trial} + \lambda_5\mathcal{L}_{cov}. \tag{12}$$

The goal loss $\mathcal{L}_{goal}$ supervises the predicted mean and log-variance through the Gaussian negative log-likelihood of the standardized goal. The reward loss $\mathcal{L}_{rew}$ trains the task statistics to support reward prediction. The attribute loss $\mathcal{L}_{attr}$ organizes task embeddings through contrastive objectives, while $\mathcal{L}_{trial}$ encourages consistency across episodes of the same trial. Finally, $\mathcal{L}_{cov}$ penalizes off-diagonal covariance in $[u_t||\mu_t]$.

Table 2: Gradient ownership in STAC. Each objective updates a disjoint set of parameters. Removing the stop-gradient (M-noIso, Sec. 5.2) additionally lets the TD loss update g_ψ.

Objective	**Updates**	**Does not update**
TD loss $\mathcal{L}_Q$	critics Q_{ν_i}, control encoder f_ϕ, adapter A_η	task encoder g_ψ
Actor loss $\mathcal{L}_\pi$	actor π_w	f_ϕ, g_ψ, A_η
Task loss $\mathcal{L}_{task}$	task encoder g_ψ, reward head R_χ	f_ϕ, A_η

Goal coordinates, object and manipulation labels, and trial identities supervise the task encoder during training. These labels are not supplied directly to the policy during action selection; the policy instead receives task features computed causally from the interaction history. Gradient routing follows Tab. 2. Appendix A details the losses, target construction, normalization, and training procedures.

5 Experimental Results

In this section we explore three empirical axes: (1) whether STAC can improve performance on seen task configurations and generalization to held-out ones; (2) which architectural choices enhance learning; and (3) what information is retained in STAC's learned control and task representations. Accordingly, we compare STAC with well established meta-RL and history-based baselines, evaluate design choices through ablations, and examine the resulting representation geometry.

Meta-World (Yu et al., 2019) is a simulated robot-arm benchmark with manipulation tasks such as reaching, pushing, and placing objects. We use its ten ML10 training task families under the hidden-goal protocol described in Appendix C. At the start of each trial, the environment samples a valid object placement and target that remain fixed throughout the trial. The physical state resets between episodes, but the agent retains its interaction history. For our experiments, goal coordinates are removed from agent observations. For each task family, we partition the position coordinate with the largest allowed range: training and in-support evaluation (A) use the lower 75%, while held-out evaluation (B) tests generalization to unseen positions. All other coordinates retain their full valid ranges. During evaluation, we use 20 configurations per task family, yielding 200 evaluation trials for splits (A) and (B) for every seed. An episode is deemed successful when Meta-World's task-specific success condition is met, such as reaching a target or moving an object to a desired position. We report the fraction of successful episodes, averaged equally across task families, using the mean and sample standard deviation over three training seeds.

5.1 ML10 Configuration Generalization

Fig. 2 compares STAC concat and STAC gating with VariBAD, AMAGO2, RL^2 with PPO, ECET, and PEARL on both configuration splits: in-support positions (A), sampled from the training region, and held-out configuration (B). Both STAC variants achieve the highest average success in both the in-support and held-out evaluations. At least one STAC variant achieves the highest per-task performance in nine of the ten task families in (A), and seven in (B). The full per-task results are tabulated in the Appendix F. The advantage of STAC therefore extends to configurations outside the training region, although the gains are not uniform across tasks.

Among the baselines, VariBAD had the highest mean performance across the in-support evaluation at $51.1 \pm 7.5\%$, while AMAGO2 performed the best on the held-out tasks with a mean of $44.1 \pm 1.6\%$. For the in-support configurations, STAC concat achieves $68.9 \pm 4.6\%$ mean success and STAC gating achieves $70.9 \pm 3.5\%$. In the held-out configuration, STAC concat and gating achieve $62.1 \pm 0.7\%, 62.2 \pm 3.0\%$ respectively. STAC's strong performance extends to challenging manipulation families such as pick-place and peg-insert-side, on which several baselines achieve little to no success. These tasks require the agent to retain a memory of the target, while coordinating grasping, object transport, and precise placement, which is well aligned with STAC's joint inference of the latent control conditions and task belief.

5.2 Architectural Ablations

Figure 3 compares encoder architectures and also examines the effects of the control encoder's window length, gradient routing, and encoder separation. STAC concat acts as the main reference

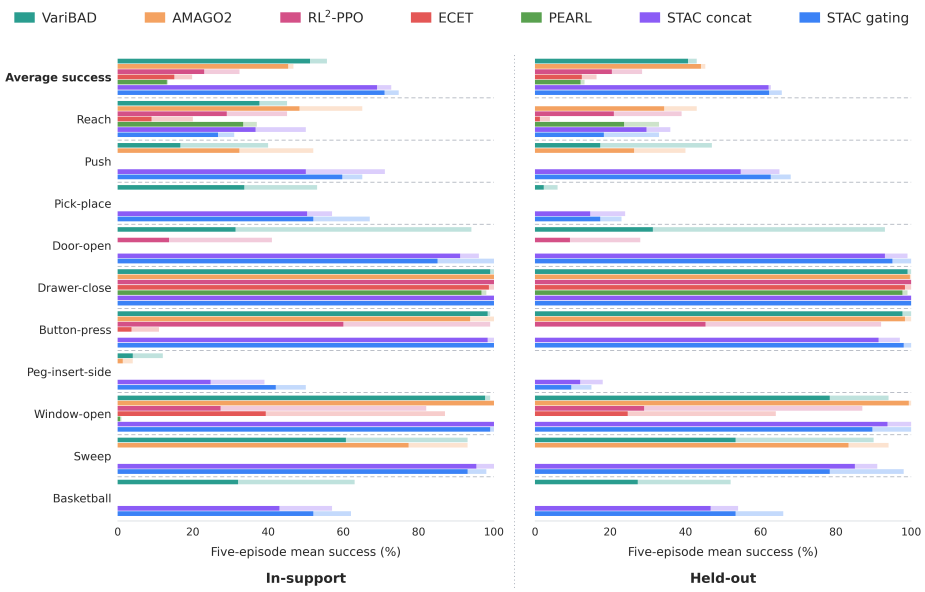


Figure 2: Mean success across ML10 tasks for STAC concat, STAC gating, and baselines, averaged over three random seeds. Left and right panels show in-support and held-out configurations.

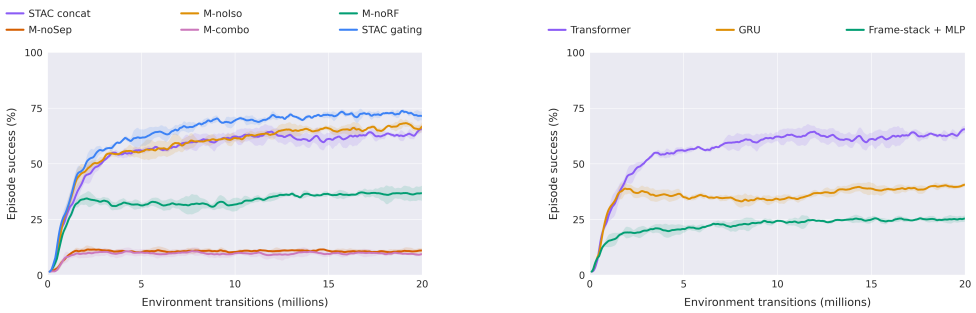
Table 3: Structural ablation configurations. The M-variants modify STAC concat. Task-path TD indicates whether critic gradients propagate from the task adapter into the task encoder; a shared trunk can still receive critic gradients through the control path. Counts include the complete agent.

Variant	Encoders	Control context	Task-path TD	Params
STAC concat	Separate	$\leqslant$ 97 tokens	Blocked	2.44M
STAC gating	Separate	$\leqslant$ 97 tokens	Blocked	2.52M
M-noIso	Separate	$\leqslant$ 97 tokens	Allowed	2.44M
M-noRF	Separate	Full trial	Blocked	2.44M
M-noSep	Shared trunk	Full trial	Blocked	2.22M
M-combo	Shared trunk	Full trial	Allowed	2.22M

for these comparisons: its control encoder can access the 97 most recent tokens, while its separate task encoder can attend to the entire trial. The corresponding results are also tabulated and provided in Tables 9 and 10 in Appendix F.

Variants that use the same encoder (i.e., shared trunk) for task and control latents perform worst: M-noSep and M-combo reach only $10.7 \pm 0.6\%$ and $9.5 \pm 1.2\%$ average success, compared to $65.4 \pm 2.6\%$ for STAC concat and $70.2 \pm 2.3\%$ for STAC gating. These variants achieve some success on the easier tasks, but struggle on those that require sustained object manipulation, suggesting that a common representation may struggle to support both persistent task inference and rapidly changing control-state estimation. However, because encoder sharing is confounded with full-history access, M-noRF more directly tests STAC's history allocation by retaining separate encoders and the same parameter count while allowing the control encoder to attend to the full trial. The resulting performance degradation supports limiting the control encoder to recent interactions, as distant interactions may aid task inference but complicate estimation of the rapidly changing control state. Allowing critic gradients into the task encoder in M-noIso yields $67.6 \pm 4.3\%$, compared with $65.4 \pm 2.6\%$ for STAC concat, and $70.2 \pm 2.3\%$ for STAC gating. This comparison does not establish a performance benefit from critic gradient isolation.

We next compare alternative architectures for the control encoder. Fig. 3b evaluates three backbones: the locally masked Transformer used by STAC concat, a windowed GRU, and a frame-stack encoder followed by an MLP. Only the control-history encoder changes; all variants retain the same Transformer-based task encoder, learning objectives, and training budget. The Transformer, GRU, and frame-stack variants contain approximately 2.44M, 2.15M, and 3.90M parameters, respectively. Despite having the most parameters, the frame-stack MLP performs worst, indicating that increased



(a) Examination of the effect of control-history length, gradient routing, and encoder sharing.

(b) Comparison of different control-encoder backbones while retaining the full-history task encoder.

Figure 3: Architectural ablations.

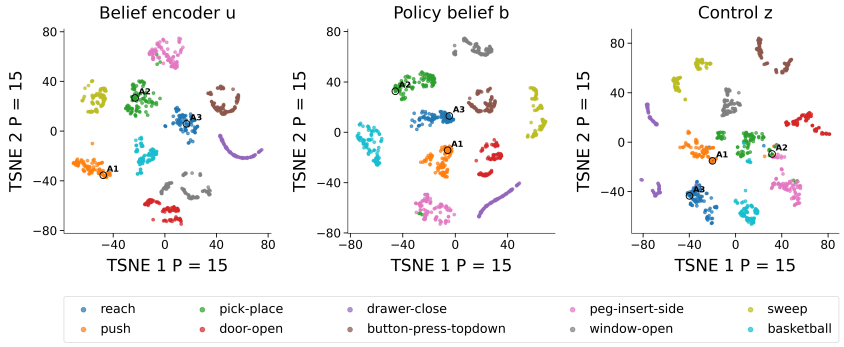


Figure 4: Task-family structure in learned representations. Each point is one evaluation trial at step 325, colored by task family; t-SNE shows stronger task-family structure in u_t, b_t than in z_t.

capacity alone is insufficient to match the Transformer's performance. The comparison therefore supports the selected Transformer architecture within the evaluated designs.

5.3 Representation Geometry and State Retrieval

To visualize the learned representations, we sample each evaluation trial at step 325 and extract (u_t, b_t, z_t) from the available history. Each point in Fig. 4 represents one trial at the same decision step rather than an entire trajectory. We normalize and project each representation separately with t-SNE, then color points by task family. The belief features form relatively distinct task-family regions, consistent with their task-attribute supervision. The control features exhibit more overlap, as similar local physical situations can arise across tasks. Thus, z_t is less strongly structured by task identity than u_t and b_t.

The pick-place anchor in Fig. 5 makes this overlap more concrete. Its two nearest neighbors in the original z_t space show similar hand and object positions across trials, whereas a random frame from another pick-place trial is much less similar. One neighbor differs from the anchor by only 5.1 mm in hand/object position despite a 117.0 mm goal difference. The nearest cross-task neighbor comes from reach yet also depicts a similar local hand-object configuration: its 41.8 mm position discrepancy is much smaller than the random same-task frame's 117.5 mm difference. These examples illustrate that z_t can place related control situations near one another across task boundaries. Limiting the control encoder's history biases z_t toward representing the current situation without strict task invariance, while the belief encoder encourages b_t to encode goal and task type. Complete projections and frame examples appear in Appendix D.

Representation geometry alone cannot reveal how the policy uses b_t. The belief interventions in Appendix E provide a behavioral check: same-goal replay leaves success nearly unchanged, different-goal substitution causes a slight reduction, and zeroing b_t leads to policy collapse. These results suggest that goal-consistent beliefs aid performance and that b_t may carry other useful signals.

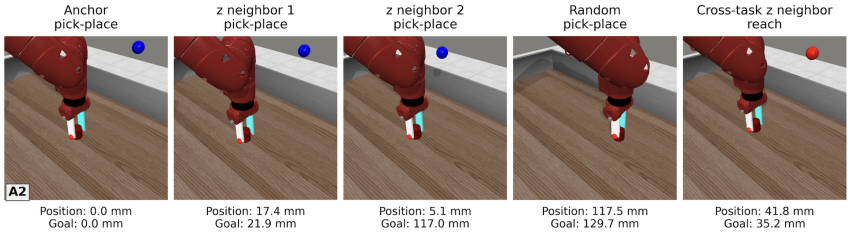


Figure 5: Nearest neighbors of a z_t pick-place anchor. The panels show the anchor, its two nearest same-task frames from other trials, a random same-task frame, and the nearest eligible cross-task frame (reach), ranked by cosine distance in the original z_t space.

6 Discussion

Earlier interactions can remain informative about the task after the physical conditions they describe have changed. Propositions 1 and 2 distinguish information available in the interaction history from information retained by a learned representation. In STAC, full-causal attention makes earlier interactions accessible to the task encoder, while dedicated inference objectives encourage retention of useful task information. STAC's empirical results support a corresponding design principle: history should be bounded for control while preserving access to longer-term evidence for task inference.

The ablations strongly support separating control and task inference in both temporal context and encoder structure. Restricting the control pathway to recent interactions raises success from 38% to 65.4%, with unchanged parameter count and training objectives, demonstrating the benefit of bounded context even when the full history contains more information. The shared-trunk results further support using separate representations for control and task inference. Gradient isolation complements this separation by ensuring that the task encoder and control encoder are both shaped by their own dedicated objectives.

The two representations can serve different roles while retaining some overlapping information. Both encode task-family structure, and neighboring control features can correspond to similar physical configurations despite different goals. Belief substitutions further establish that the trained policy is sensitive to its task input (Appendix E). Strong performance with both concatenation and gating STAC variants further suggests that this organization can support different mechanisms for conditioning decisions on task information.

The current evidence is limited to generalization within fixed task families and privileged task supervision during training. A broader future direction is to learn history allocation dynamically, extending control context when older events remain relevant and revising task memory when tasks change. Evaluating such mechanisms on new task families and without privileged task supervision would test whether the same design principle extends to settings where relevant temporal horizons and task specifications are not explicitly provided during training.

7 Conclusion

This work asks how agents should represent interaction histories containing information that evolves over different timescales. We addressed this question through State-Task Adaptive Control (STAC), which assigns recent historical context to a control representation while using the complete trial to construct a task-belief representation. Comparisons with history-conditioned RL baselines and controlled architectural variants show that this structured allocation of history improves learning and generalization across heterogeneous manipulation tasks. The ablations further show that the proposed design choices materially affect performance. Representation analyses additionally characterize how task evidence accumulates and how information is distributed between the two pathways. Together, these results show that effective memory design depends not only on how much history an agent retains, but also on which representations access that history and how they are trained.

AI Assistance Statement

Generative AI tools were used for language editing and proofreading, including improvements to clarity, grammar, and typographical errors, and for assistance with code revision. All scientific content, ideas, analyses, interpretations, and conclusions were produced and verified by the authors, and all AI-assisted text and code revisions were manually reviewed and verified. The authors take full responsibility for the final content of the manuscript.

References

Kei Akuzawa, Yusuke Iwasawa, and Yutaka Matsuo. Estimating disentangled belief about hidden state and hidden task for meta-reinforcement learning. In *Learning for Dynamics and Control*, pp. 73–86. PMLR, 2021.

Era Choshen and Aviv Tamar. ContraBAR: Contrastive bayes-adaptive deep rl, 2023. URL https://arxiv.org/abs/2306.02418.

Finale Doshi-Velez and George Konidaris. Hidden parameter markov decision processes: A semiparametric regression approach for discovering latent task parametrizations. In *IJCAI: proceedings of the conference*, volume 2016, pp. 1432, 2016.

Yan Duan, John Schulman, Xi Chen, Peter L. Bartlett, Ilya Sutskever, and Pieter Abbeel. RL^2: Fast reinforcement learning via slow reinforcement learning. *CoRR*, abs/1611.02779, 2016. URL http://arxiv.org/abs/1611.02779.

Jake Grigsby, Linxi Fan, and Yuke Zhu. AMAGO: Scalable in-context reinforcement learning for adaptive agents, 2024a. URL https://arxiv.org/abs/2310.09971.

Jake Grigsby, Justin Sasek, Samyak Parajuli, Daniel Adebi, Amy Zhang, and Yuke Zhu. AMAGO-2: Breaking the multi-task barrier in meta-reinforcement learning with transformers, 2024b. URL https://arxiv.org/abs/2411.11188.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. *CoRR*, abs/1801.01290, 2018. URL http://arxiv.org/abs/1801.01290.

Jan Humplik, Alexandre Galashov, Leonard Hasenclever, Pedro A Ortega, Yee Whye Teh, and Nicolas Heess. Meta reinforcement learning as task inference. *arXiv preprint arXiv:1905.06424*, 2019.

Prannay Khosla, Piotr Teterwak, Chen Wang, Aaron Sarna, Yonglong Tian, Phillip Isola, Aaron Maschinot, Ce Liu, and Dilip Krishnan. Supervised contrastive learning. *CoRR*, abs/2004.11362, 2020. URL https://arxiv.org/abs/2004.11362.

Luckeciano C. Melo. Transformers are meta-reinforcement learners, 2022. URL https://arxiv.org/abs/2206.06614.

Tianwei Ni, Benjamin Eysenbach, and Ruslan Salakhutdinov. Recurrent model-free RL is a strong baseline for many pomdps. *CoRR*, abs/2110.05038, 2021. URL https://arxiv.org/abs/2110.05038.

Kate Rakelly, Aurick Zhou, Deirdre Quillen, Chelsea Finn, and Sergey Levine. Efficient off-policy meta-reinforcement learning via probabilistic context variables. *CoRR*, abs/1903.08254, 2019. URL http://arxiv.org/abs/1903.08254.

Gresa Shala, André Biedenkapp, Pierre Krack, Florian Walter, and Josif Grabocka. Efficient cross-episode meta-rl. In Y. Yue, A. Garg, N. Peng, F. Sha, and R. Yu (eds.), *International Conference on Learning Representations*, volume 2025, pp. 75347–75388, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/file/bbc0df00853596fcf4bbcbef853a880a-Paper-Conference.pdf.

Jianlin Su, Yu Lu, Shengfeng Pan, Bo Wen, and Yunfeng Liu. Roformer: Enhanced transformer with rotary position embedding. *CoRR*, abs/2104.09864, 2021. URL https://arxiv.org/abs/2104.09864.

Ruibin Xiong, Yunchang Yang, Di He, Kai Zheng, Shuxin Zheng, Chen Xing, Huishuai Zhang, Yanyan Lan, Liwei Wang, and Tie-Yan Liu. On layer normalization in the transformer architecture. *CoRR*, abs/2002.04745, 2020. URL https://arxiv.org/abs/2002.04745.

Denis Yarats, Amy Zhang, Ilya Kostrikov, Brandon Amos, Joelle Pineau, and Rob Fergus. Improving sample efficiency in model-free reinforcement learning from images. *CoRR*, abs/1910.01741, 2019. URL http://arxiv.org/abs/1910.01741.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-world: A benchmark and evaluation for multi-task and meta reinforcement learning. *CoRR*, abs/1910.10897, 2019. URL http://arxiv.org/abs/1910.10897.

Amy Zhang, Shagun Sodhani, Khimya Khetarpal, and Joelle Pineau. Learning robust state abstractions for hidden-parameter block mdps. *arXiv preprint arXiv:2007.07206*, 2020.

Luisa Zintgraf, Kyriacos Shiarlis, Maximilian Igl, Sebastian Schulze, Yarin Gal, Katja Hofmann, and Shimon Whiteson. Varibad: A very good method for bayes-adaptive deep rl via meta-learning, 2020. URL https://arxiv.org/abs/1910.08348.

A Additional method details

A.1 Temporal indexing, receptive fields, and replay construction

Figure 6 illustrates how the interaction stream is converted into the causal tokens supplied to both encoders. Each trial contains five episodes of 150 decision steps. At position t, the agent combines its current observation with the action and reward from preceding transitions:

$$\xi_t = [o_t || a_{t-1} || \operatorname{symlog}(r_{t-1}) || c_t] \in \mathbb{R}^{48}. \tag{A.1}$$

The bottom panel shows the token's components: the 39-dimensional observation, with its goal coordinates set to zero; the four-dimensional previous action; the scalar transformed reward; and four temporal features. Using the preceding action and reward ensures that every component is available before the agent selects a_t.

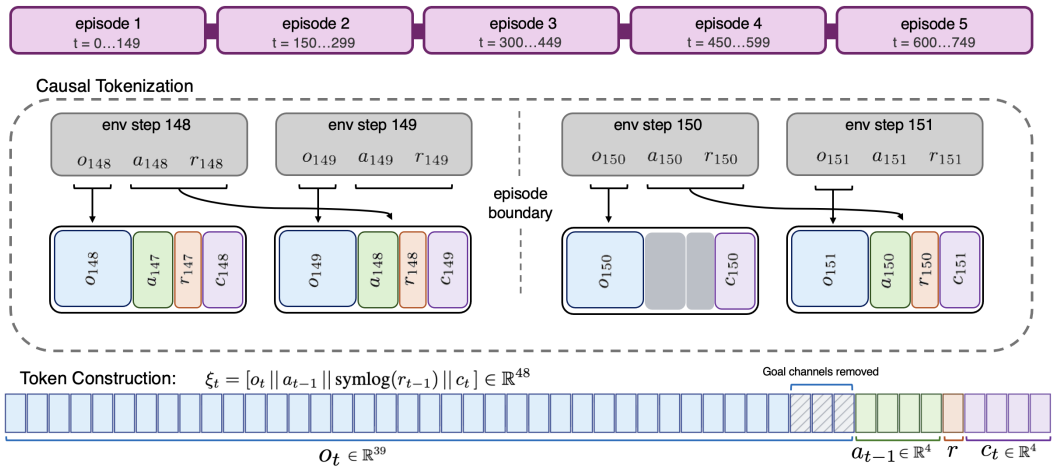


Figure 6: Causal token construction across episode boundaries. Each token combines the current observation, previous action, transformed previous reward, and temporal features.

The middle panel illustrates token construction around an episode boundary. Within an episode, each new token incorporates the observation resulting from the preceding action. At an internal reset, the observation is replaced by the initial observation of the next episode, while the preceding action and reward are retained. For example, the first token of the second episode is

$$\xi_{150} = [o_{150}^{reset} || a_{149} || \operatorname{symlog}(r_{149}) || c_{150}], \tag{A.2}$$

Each encoder embeds tokens with its own two-layer MLP, $e_t^{ctrl} = E_{ctrl}(\xi_t)$ and $e_t^{task} = E_{task}(\xi_t)$, and processes the embedded sequence with a pre-LayerNorm transformer (Xiong et al. 2020) using rotary position embeddings (Su et al. 2021). The two transformer stacks T_ϕ, T_ψ have identical configurations with the exception of their causal masks. The output of the last transformer layer for each encoder is:

$$h_t^{ctrl} = T_\phi(e_{0:t}^{ctrl}; M_W)_t, \quad h_t^{task} = T_\psi(e_{0:t}^{task}; M_\infty)_t. \tag{A.3}$$

The control state is $z_t = \operatorname{LayerNorm}(W_z h_t^{ctrl})$, and three linear heads produce the task statistics from Eq. 9:

$$u_t = \frac{W_u h_t^{task}}{||W_u h_t^{task}||_2}, \quad \mu_t = W_\mu h_t^{task}, \quad \ell_t = \operatorname{clip}(W_\ell h_t^{task}, -10, 2). \tag{A.4}$$

Replay stores complete trials rather than isolated transitions. Each learning update samples $B = 16$ trials with replacement, approximately balanced across the available task families. Online and target representations are recomputed from the stored tokens. For TD learning, we independently sample 256 positions per trial, uniformly with replacement from $\{0, ..., T-1\}$.

During action selection, the control representation is recomputed from the most recent $\min(t+1, 97)$ tokens. The task encoder maintains a causal key-value cache throughout the trial. When learning changes its parameters, this cache is rebuilt from the accumulated raw tokens before subsequent action selection. Thus, cached representations do not retain activations computed under outdated encoder parameters. All trial histories and caches are cleared when a new trial begins.

A.2 TASK-REPRESENTATION OBJECTIVES

The task encoder produces a unit-normalized embedding u_t, a standardized-goal mean μ_t, and a diagonal log-variance ℓ_t. Its objective is

$$\mathcal{L}_{task} = \mathcal{L}_{goal} + \mathcal{L}_{rew} + 0.3\mathcal{L}_{attr} + 0.1\mathcal{L}_{trial} + 0.1\mathcal{L}_{cov}. \tag{A.5}$$

These losses train the task encoder and its auxiliary prediction heads. They do not update the control encoder or the policy-facing adapter.

Goal prediction. Let $g_i \in \mathbb{R}^3$ denote the target position of trial i. We standardize targets using

$$\tilde{g}_{i,d} = \frac{g_{i,d} - m_{g,d}}{s_{g,d}}, \quad \text{for } d = 1, 2, 3 \tag{A.6}$$

where m_g and s_g are coordinate-wise statistics estimated from 1,000 valid training-support configurations per task family. A single set of statistics is shared across families and remains fixed during learning and evaluation. Standard deviations are floored at 10^{-3}.

For each sampled trial, we draw 128 positions uniformly with replacement. Let $\mathcal{S}$ denote the resulting collection of trial-position pairs. The goal objective is the diagonal-Gaussian negative log-likelihood:

$$\mathcal{L}_{goal} = \frac{1}{|\mathcal{S}|} \sum_{(i,t)\in\mathcal{S}} \frac{1}{2} \sum_{d=1}^{3} \left[(\tilde{g}_{i,d} - \mu_{i,t,d})^2 e^{-\ell_{i,t,d}} + \ell_{i,t,d} \right]. \tag{A.7}$$

The predicted log-variance is clipped to $[-10, 2]$.

Reward prediction. For each step t, we use the available task statistics to predict rewards for observation-action pairs at other positions j in the same trial:

$$\hat{r}_{i,t,j} = R_\chi([u_{i,t}||\mu_{i,t}||\ell_{i,t}||o_{i,j}||a_{i,j}]). \tag{A.8}$$

We draw four query positions uniformly from $j < t$ and four from $j > t$, with replacement. At $t = 0$, the four unavailable earlier queries use $j = 1$, and at $t = T - 1$, the four missing queries use $j = T - 2$. The corresponding loss is:

$$\mathcal{L}_{rew} = \frac{1}{8|\mathcal{S}|} \sum_{(i,t)\in\mathcal{S}} \sum_{j\in\mathcal{J}_{i,t}} \left[\hat{r}_{i,t,j} - \text{symlog}(r_{i,j}) \right]^2. \tag{A.9}$$

This objective trains the task statistics to support reward prediction across the trial. Future query observations and actions are supplied to the auxiliary predictor during training; the task statistics themselves depend only on the history available at t.

Attribute contrastive learning. We apply this objective to the task embedding at the final decision position for each trial in the batch. For B trials, let

$$v_i = \frac{u_{i,T-1}}{||u_{i,T-1}||_2}, \quad V = \{v_i\}_{i=1}^{B}. \tag{A.10}$$

Given a set of labels y, examples with the same label form positive pairs. For anchor i, let $P_y(i) = \{p \neq i : y_p = y_i\}$. The supervised contrastive (Khosla et al. 2020) loss is

$$\text{SupCon}(V; y) = -\frac{1}{|\mathcal{A}_y|} \sum_{i\in\mathcal{A}_y} \frac{1}{|P_y(i)|} \sum_{p\in P_y(i)} \log \frac{\exp(v_i^T v_p/\tau_c)}{\sum_{j\neq i} \exp(v_i^\top v_j/\tau_c)}. \tag{A.11}$$

Here, $\mathcal{A}_y$ contains anchors with at least one positive and one negative in the batch, and $\tau_c = 0.1$. If there are no valid anchors, we set the loss to zero. We use two sets of labels derived from the underlying task. The *object* labels identify the target, while *verb* labels identify the agent action

behavior. For example, door-open and window-open have different object labels but both share the same verb *open*. We apply the SupCon objective to each object and manipulation label:

$$\mathcal{L}_{attr} = \frac{1}{2}\,\mathrm{SupCon}(V; y^{object}) + \frac{1}{2}\,\mathrm{SupCon}(V; y^{verb}). \tag{A.12}$$

Cross-episode consistency. The task remains fixed across episode resets, so we encourage task features from different episodes within the same trial to agree. For each sampled trial, we choose one position in each of two different episodes, sampling uniformly from the second half of each episode. One produces an online query and the other produces a detached key from the exponential-moving-average task network:

$$q_i = \mathrm{norm}([u_{i,t_i}||\mu_{i,t_i}]), \quad k_i = \mathrm{sg}[\mathrm{norm}([\bar{u}_{i,t'_i}||\bar{\mu}_{i,t'_i}])] \tag{A.13}$$

The query is matched with keys from the same trial and contrasted with keys from other trials. If a trial appears more than once in a batch, it can provide multiple positive keys. Writing P_i for the positive keys and $\mathcal{A}_{trial}$ for queries with both positive and negative keys, the corresponding loss is

$$\mathcal{L}_{trial} = -\frac{1}{|\mathcal{A}_{trial}|}\sum_{i\in\mathcal{A}_{trial}}\frac{1}{|P_i|}\sum_{j\in P_i}\log\frac{\exp(q_i^\top k_j/\tau_c)}{\sum_{l=1}^{B}\exp(q_i^\top k_l/\tau_c)} \tag{A.14}$$

Covariance regularization. Let $x_{i,t} = [u_{i,t}||\mu_{i,t}] \in \mathbb{R}^{35}$ and let C be its sample covariance over $\mathcal{S}$. We penalize its off-diagonal entries through

$$\mathcal{L}_{cov} = \frac{1}{35}\sum_{a\neq b} C_{ab}^2. \tag{A.15}$$

This discourages covariance among task features and goal predictions. It does not impact the control representation or prevent u_t from containing goal information.

Table 4: STAC hyperparameters, shared by STAC concat and STAC gating unless indicated

Setting	Value
Trial and tokens	
Episodes per trial; steps per episode	$5; 150$
Trial length T	750
Token dimension (obs, act, rew, time)	$48 = 39 + 4 + 1 + 4$
Encoder backbones (separate parameters)	
Embedding MLP layers; hidden width; output width	$2; 128; 128$
Transformer layers D; attention heads	$3; 4$
Model width; feed-forward width	$128; 512$
Normalization; positional encoding	pre-LayerNorm; rotary
Attention band W	32 (for control encoder), ∞ (for task encoder)
Representation	
Control state z_t dimensions	64
Task statistics $v_t = [u_t \|\| \mu_t \|\| \ell_t]$	$38 = 32 + 3 + 3$
Log-variance range for ℓ_t	$[-10, 2]$
Adapter hidden width; task belief b_t dimension	$64; 32$
Actor, critics, and reward predictor	
Actor and critic hidden layers; width; activation	$3; 400$; ReLU
Reward predictor hidden layers; width; activation	$2; 256$; ReLU
Trainable network parameters (STAC concat; STAC gating)	2.44M; 2.52M
Task-inference objectives	
$\lambda_{goal}, \lambda_{rew}, \lambda_{attr}, \lambda_{trial}, \lambda_{cov}$	$1, 1, 0.3, 0.1, 0.1$
Contrastive temperature τ_c	0.1
Positions per trial for $\mathcal{L}_{goal}, \mathcal{L}_{rew}, \mathcal{L}_{cov}$	128
$\mathcal{L}_{trial}$ EMA	0.005
Optimization and replay	
Trials per batch B; TD positions per trial	$16; 256$
Discount γ	0.99
Optimizers	Adam
Learning rate (all parameter groups)	3×10^{-4}
Target update rate	0.005
PopArt EMA rate; standard-deviation floor	$3 \times 10^{-4}; 0.05$

B THEORY AND PROOFS

B.1 ADDITIONAL INTERACTIONS PROVIDE NONNEGATIVE EXPECTED TASK INFORMATION

We will assess the information available for inferring the agent's task and distinguish it from the information retained by the learned task representation. First note the history available before selecting action a_t,

$$H_t = (o_0, a_0, r_0, o_1, ..., a_{t-1}, r_{t-1}, o_t), \tag{B.1}$$

after choosing a_t the agent observes (r_t, o_{t+1}) and the updated history becomes $H_{t+1} = (H_t, a_t, r_t, o_{t+1})$. The task Θ is fixed throughout a trial, including across episode resets. The exact joint posterior and learned representations are

$$B_t(x, \theta) = p(X_t = x, \Theta = \theta | H_t), \quad z_t = f_\phi(H_t^{(K)}), \quad b_t = A_\eta(g_\psi(H_t)), \quad s_t = [z_t || b_t] \tag{B.2}$$

B_t is the joint posterior, while $p(\theta|H_t)$ denotes the task marginal. The vector b_t is a learned representation and is not assumed to equal either distribution. For this analysis, we fix the encoder parameters. Actions depend on the observed history and on randomization that is independent of the task.

Before analyzing the learned task representation, we first characterize the task evidence available in the interaction history itself. The task Θ remains fixed during a trial and H_{t+1} contains H_t, the following proposition shows that the mutual information between the task and the complete history cannot decrease as new interactions are added. This concerns information available in the history; it does not imply that a learned representation $b_t = A_\eta(g_\psi(H_t))$ necessarily preserves every piece of information.

Proposition 1 (Accumulation of Available Task Information)**.** *Under the assumptions above,*

$$\begin{aligned} I(\Theta; H_{t+1}) - I(\Theta; H_t) &= I\big(\Theta; (a_t, r_t, o_{t+1}) \mid H_t\big) \\ &= I\big(\Theta; (r_t, o_{t+1}) \mid H_t, a_t\big) \\ &= \mathbb{E}_{H_{t+1}}[D_{\mathrm{KL}}(p(\theta \mid H_{t+1}) \,\|\, p(\theta \mid H_t))] \geq 0. \end{aligned} \tag{B.3}$$

The increment is strictly positive if and only if the updated posterior differs from the previous posterior on a set of histories with positive probability. Equivalently, the next reward and observation must contain information about Θ, conditional on the existing history and selected action a_t.

Proof. Since $H_{t+1} = (H_t, a_t, r_t, o_{t+1})$, the histories are nested, so the chain rule gives

$$\begin{aligned} I(\Theta; H_{t+1}) &= I(\Theta; (H_t, a_t, r_t, o_{t+1})) \\ &= I(\Theta; H_t) + I(\Theta; (a_t, r_t, o_{t+1}) | H_t) \end{aligned} \tag{B.4}$$

Subtracting $I(\Theta; H_t)$ from both sides yields the first equality in Eq. B.3.

Next, applying the conditional chain rule to the new interaction (a_t, r_t, o_{t+1}) gives

$$I(\Theta; (a_t, r_t, o_{t+1}) | H_t) = I(\Theta; a_t | H_t) + I(\Theta; (r_t, o_{t+1}) | H_t, a_t). \tag{B.5}$$

Because the policy uses only the observed history and independent randomization,

$$p(a_t | H_t, \Theta) = p(a_t | H_t), \quad \text{Therefore } I(\Theta; a_t | H_t) = 0 \tag{B.6}$$

Although a_t may be marginally correlated with Θ through the task-dependent history H_t, it provides no additional task information once the complete history is given. Substituting this result into Eq. B.5 gives

$$I(\Theta; (a_t, r_t, o_{t+1}) | H_t) = I(\Theta; (r_t, o_{t+1}) | H_t, a_t), \tag{B.7}$$

which establishes the second equality in Eq. B.3. Finally, applying the posterior form of conditional mutual information gives

$$\begin{aligned} I(\Theta; (a_t, r_t, o_{t+1}) | H_t) &= \mathbb{E}_{\Theta, H_{t+1}} \left[\log \frac{p(\Theta | H_{t+1})}{p(\Theta | H_t)} \right] \\ &= \mathbb{E}_{H_{t+1}} \left[\int p(\theta | H_{t+1}) \log \frac{p(\theta | H_{t+1})}{p(\theta | H_t)} d\theta \right] \\ &= \mathbb{E}_{H_{t+1}} [D_{KL}(p(\theta | H_{t+1}) || p(\theta | H_t))]. \end{aligned} \tag{B.8}$$

For a collection of discrete tasks, the integral is replaced by a sum over different θ. Since KL divergence is nonnegative,

$$D_{KL}\left(p(\theta | H_{t+1}) || p(\theta | H_t)\right) \geq 0, \tag{B.9}$$

for every realized history update. Taking the expectation of Eq. B.9 remains nonnegative, and it is nonzero as long as $p(\theta | H_{t+1}) \neq p(\theta | H_t)$.

Thus, the increment is strictly positive precisely when the new reward and observation change the task posterior on a set of positive probability. The vanishing action term does not mean that action selection does not impact information acquisition, because the action influences the task-dependent likelihood through $p(r_t, o_{t+1} | H_t, a_t, \theta)$ and therefore determines which evidence is available in the

next reward and observation. The result consequently permits adaptive actions and correlated transitions; it does not require interactions to be independent.

In each trial, an episode reset changes the distribution of the physical state but does not invalidate previously collected evidence, provided that Θ remains fixed and the earlier interactions remain in H_{t+1}. Proposition 1 therefore also applies to transitions that contain an episode reset. □

When the relevant task entropies are finite, Eq. B.3 can be rewritten as

$$H(\Theta|H_{t+1}) \leq H(\Theta|H_t). \tag{B.10}$$

This is an average statement over possible next interactions, as the entropy of the posterior may increase after a particular observation, even though it cannot increase on average.

Finally, because $b_t = A_\eta(g_\psi(H_t))$ is a function of the history, the variables form the following Markov chain:

$$\Theta \to H_t \to b_t. \tag{B.11}$$

The data-processing inequality therefore gives

$$I(\Theta; b_t) \leq I(\Theta; H_t). \tag{B.12}$$

However, Proposition 1 does not suggest that

$$I(\Theta; b_{t+1}) \geq I(\Theta; b_t). \tag{B.13}$$

A finite-capacity or poorly trained encoder may discard previously available task evidence even through the information contained in the complete history is nondecreasing. This gap between the task information available in the history and evidence retained by a learned representation is a central motivating challenge behind STAC. The task-belief encoder attends to the entire agent history, while the inference objectives encourage b_t to retain as much task-relevant information as possible. These design choices do not theoretically guarantee monotonic information retention; instead our representation probes and controlled ablations examine whether they improve the accumulation and preservation of task information in practice.

B.2 Latent Representations as Control-Oriented Summaries

Proposition 1 characterizes the task information available in the complete interaction history. We next distinguish this available information from the information retained by the learned agent state. Recall that

$$B_t(x, \theta) = p(X_t = x, \Theta = \theta \mid H_t) \tag{B.14}$$

is the exact joint posterior, whereas

$$z_t = f_\phi(H_t^{(K)}), \qquad b_t = g_\psi(H_t), \qquad s_t = [z_t \| b_t] \tag{B.15}$$

are learned finite-dimensional summaries. To quantify the posterior information discarded by s_t, we define

$$\varepsilon_t^{\text{post}} := I\big((X_t, \Theta); H_t \mid s_t\big). \tag{B.16}$$

This expression measures how much additional information the complete history reveals about the latent physical state and task after the learned state representation is known. For a realized representation s,

$$p(X_t = x, \Theta = \theta | S_t = s) = \mathbb{E}\left[B_t(x, \theta) | S_t = s\right]. \tag{B.17}$$

This expectation is taken over the conditional distribution of histories mapped to s. In the event that multiple histories are mapped to the same learned state $S_t = s$, the agent can no longer determine which history actually occurred. Consequently $p(X_t, \Theta | S_t = s)$ combines the beliefs obtained from all such histories, while weighting each history by how likely it is given $S_t = s$. Proposition 2 quantifies the information lost through this averaging as the expected KL divergence between the posterior available from the complete history and the posterior available from S_t alone. This loss is zero exactly when all histories mapped to the same representation induce the same posterior over (X_t, Θ).

Proposition 2 (Posterior sufficiency of the learned agent state)**.** *Suppose that s_t is a deterministic function of H_t. Then*

$$\epsilon_t^{post} = \mathbb{E}_{H_t}\left[D_{KL}\left(p(x,\theta|H_t)||p(x,\theta|s_t)\right)\right] \geq 0. \tag{B.18}$$

Moreover, $\epsilon_t^{post} = 0$ if and only if

$$p(x,\theta|H_t) = p(x,\theta|s_t) \tag{B.19}$$

Proof. Similar to Proposition 1 we can apply the posterior form of conditional mutual information to get

$$I((X_t,\Theta);H_t|s_t) = \mathbb{E}_{H_t,s_t}\left[D_{KL}(p(x,\theta|H_t,s_t)||p(x,\theta|s_t))\right]. \tag{B.20}$$

Because s_t is computed deterministically from H_t through $s_t = [f_\phi(H_t^{(K)})||A_\eta(g_\psi(H_t))]$, once the complete history is known, additionally conditioning on s_t provides no new information. Therefore,

$$p(x,\theta|H_t,s_t) = p(x,\theta|H_t). \tag{B.21}$$

Substituting Eq. B.21 into Eq. B.20 yields the result in Proposition. 2. The non-negativity of KL divergence ensures that $\epsilon_t^{\text{post}} \geq 0$. □

Proposition 1 and 2 distinguish information availability in the agent history from the information that is retained in a learned latent state s_t. Proposition 1 shows that additional interactions can make more task information available in H_t while Proposition 2 characterizes how much information relevant to (X_t,Θ) is lost when H_t is compressed into s_t. If histories mapped to the same s_t induce different posteriors, the learning representation has discarded information relevant to state or task inference. If they induce the same posterior, distinguishing among these histories would not improve inference, and no posterior information would have been lost.

Posterior sufficiency requires S_t to preserve every distinction in H_t that changes the inferred distribution over (X_t,Θ). This is stronger than what is necessary for optimal control, which requires preserving only distinctions that can change the value of an action. In a partially observable control problem, the exact belief B_t summarizes all of the information in H_t that is required to predict future agent outcomes. Consequently, at a specific decision step, if two histories induce the same belief B_t they effectively assign the same optimal value to every action. Equivalently,

$$Q_t^*(H_t,a) = \bar{Q}_t^*(B_t,a), \tag{B.22}$$

where the first term evaluates the action-value of a while using the complete history as a reference, while the second term evaluates the same action using the resulting belief. If Eq. B.19 holds, then B_t can be recovered from s_t and there exists a function $\tilde{Q}_t^*$ such that

$$Q_t^*(H_t,a) = \tilde{Q}_t^*(s_t,a) \quad \text{for every action } a \tag{B.23}$$

Posterior sufficiency therefore implies that replacing H_t with s_t does not change any optimal action-value. The converse, however, need not hold. Two histories may induce different posteriors while producing identical action values if their differences involve latent variation that does not impact future rewards or action-dependent dynamics. Eq. B.23 is therefore a weaker, control-oriented notion of sufficiency: s_t must preserve the distinctions needed to evaluate actions meaningfully, instead of trying to encode every detail about the task or environment that can be extracted from H_t

STAC is in turn designed to encourage this control-oriented information preservation, although it does not theoretically guarantee Eq. B.23. The temporal-difference objective encourages z_t to retain information useful for value estimation, while the task-inference objectives encourage b_t to retain the most important task evidence. STAC does not directly minimize ϵ_t^{post}, and $\epsilon_t^{\text{post}} > 0$ establishes that some latent information has been lost, but not that this discarded information matters for control. This distinction motivates our experiments to determine to what extent retained information supports effective decisions, which are measured through standard RL returns and task success.

B.3 TEMPORAL VIEWS

Propositions 1 and 2 distinguish the information available in the complete history from the information retained by a learned representation. A separate architectural question is which portions of the history should be made available to each encoder. Because $H_t^{(K)}$ is a suffix of H_t, the chain rule for any random variable Y becomes

$$I(Y; H_t) = I(Y; H_t^{(K)}) + I(Y; H_t | H_t^{(K)}) \geq I(Y; H_t^{(K)}). \tag{B.24}$$

Thus, the complete history may contain information that is unavailable from the most recent K interactions alone. In STAC, the task-belief encoder $b_t = A_\eta(g_\psi(H_t))$ can access this earlier evidence, whereas the control-state encoder $z_t = f_\phi(H_t^{(K)})$ cannot. Correspondingly, the data-processing inequality gives

$$I(Y; b_t) \leq I(Y; H_t), \quad I(Y; z_t) \leq I(Y; H_t^{(K)}). \tag{B.25}$$

These inequalities characterize a gap between available evidence and what is retained by the learned encoders. Full-history access does not guarantee that b_t preserves all task-relevant evidence, and restricting z_t to a recent window does not guarantee that it contains no task information, since recent observations and rewards may reveal details about the task.

C META-WORLD COMPARISON PROTOCOL

We adapt reference implementations to the common protocol in Table 5, matching environment interactions while retaining method-specific objectives, normalization, and update schedules.

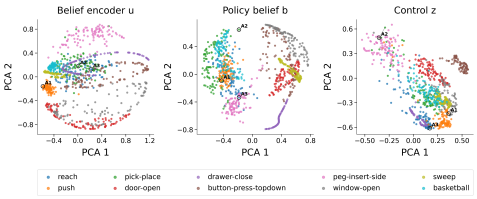
Table 5: Meta-World experimental details.

Setting	**Shared protocol**
Benchmark	Ten ML10 training families; Meta-World 3.1.1, V3 environment, v2 dense rewards.
Trials	Five 150-step episodes per fixed configurations; history/context persists across episodes and resets between trials.
Training splits	Lower 75% of each family's largest-range reset-vector axis for training/in-support evaluation, remaining 25% for held-out evaluation. Other coordinates retain their entire ranges.
Training duration	20,025,000 transitions per method/seed.
Evaluation	20 fresh configurations per task family/split with matched configurations streams; weights and learned normalization frozen.
Evaluation metric	Episode success at any step, averaged over all five episodes and equally over task families. Mean $\pm$ sample standard deviation are taken across seeds.
Method	**Application and learning schedule**
STAC concat/gating	Separate control/task encoders, with SAC-style learning
AMAGO-2	Off-policy Transformer with 256-step context, advantage-filtered actor and symlog two-hot critic; 1,000 replay updates per round.
ECET	Intra/cross-episode Transformer memories with horizon and episode-bank updates adapted to 150-step episodes; 10 PPO epochs per round.
PEARL	SAC with a probabilistic task embedding inferred from accumulated trial experience, updated between episodes; $2,000$ replay updates per round.
VariBAD	Recurrent variational belief with reward reconstruction and KL regularization, retained across episodes; 16 PPO epochs and one VAE update per round, without a privileged-task decoder.
RL^2-PPO	Gaussian GRU policy where hidden state persists across episodes; 10 PPO epochs per round.

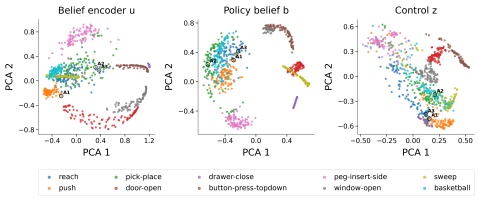
D COMPLETE REPRESENTATION FIGURES

We collect fresh evaluation rollouts from three frozen STAC checkpoints and extract (u_t, b_t, z_t) once per trial at $(t = 325)$. The PCA and t-SNE figures show the in-support and held-out configurations pools separately; both contain the same ten task families. Each feature type is normalized and projected separately. PCA shows leading directions of linear variation, while t-SNE emphasizes local neighborhoods. To check task-family grouping without a projection, we measure the fraction of each held-out trial's five nearest neighbors that share its task family: $97.57 \pm 0.54\%$ for z_t and $98.19 \pm 0.65\%$ for b_t.

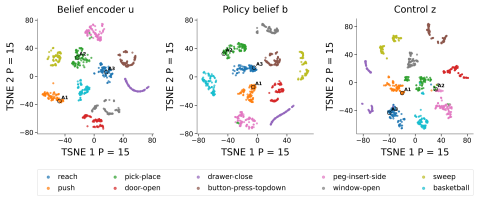
The frame galleries in Figures 8 and 9 inspect individual samples without applying projections. For each anchor, same-task neighbors come from other trials at the matched step; a random eligible same-task frame provides a comparison, and the cross-task column shows the nearest eligible frame from another task. The nearest z_t neighbors qualitatively show similar hand and object configurations even when their goals differ. This result supports the major intuition behind STAC, the latent-control neighborhoods are organized around the current control situation rather than the exact goal identity; STAC delegates this responsibility to the full-history belief b_t.



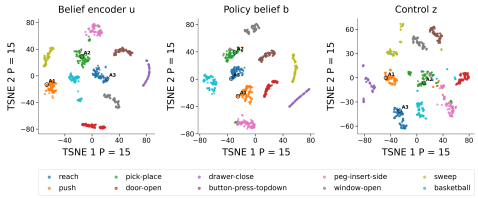
(a) PCA: in-support configurations.



(b) PCA: held-out configurations.



(c) t-SNE: in-support configurations.



(d) t-SNE: held-out configurations.

Figure 7: Representation geometry across evaluation configurations. Each point is one fresh evaluation trial at step 325, colored by task family; u_t, b_t, z_t are normalized and projected separately with perplexity 15 for t-SNE.

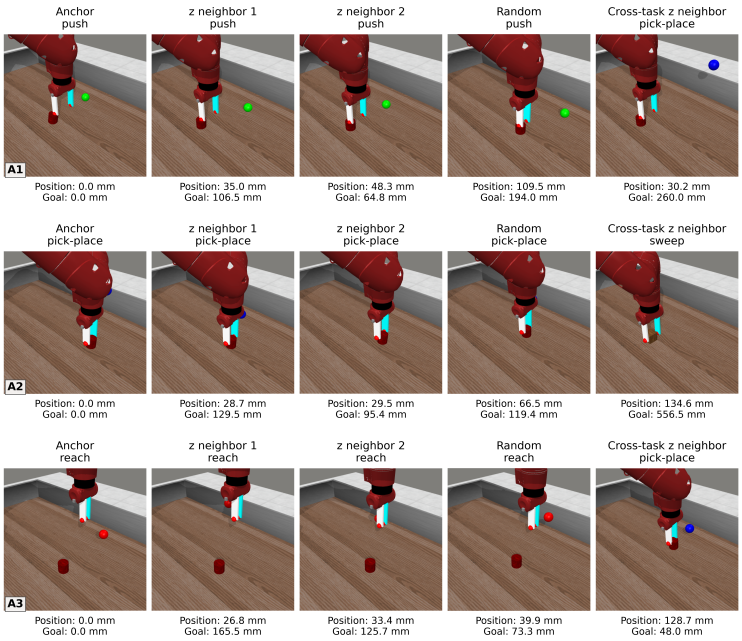


Figure 8: Control-feature neighbors on **in-support** configurations. Each row shows a push, pick-place, or reach anchor with two nearest same-task frames from other trials, a random same-task frame, and the nearest eligible cross-task frame; retrieval uses cosine distance in the original z_t space, and labels give position and goal offsets.

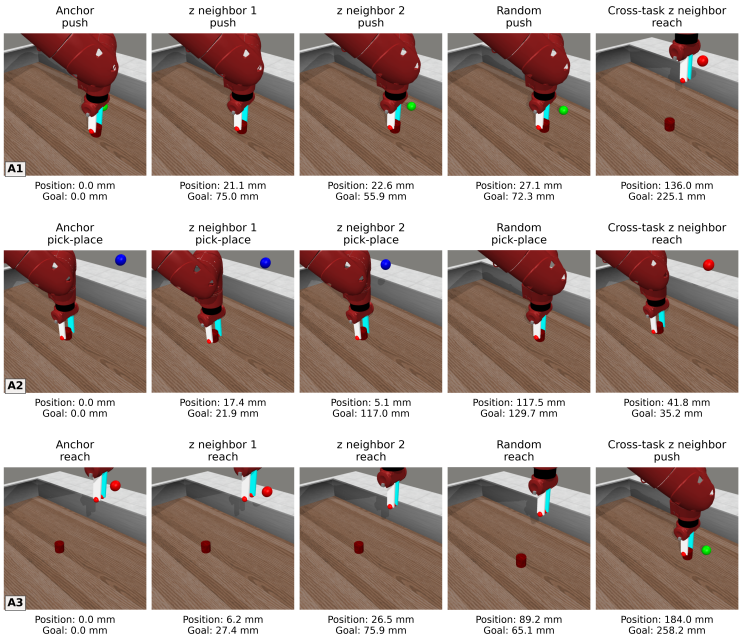


Figure 9: Control-feature neighbors on **held-out** configurations. Each row shows a push, pick-place, or reach anchor with two nearest same-task frames from other trials, a random same-task frame, and the nearest eligible cross-task frame; retrieval uses cosine distance in the original z_t space, and labels give position and goal offsets.

E BELIEF INTERVENTIONS

The results in Table 6 show that replacing the belief with one from a different history but the same goal leaves success nearly unchanged for both training-support and held-out configurations. A different goal-belief lowers success modestly, while removing the belief altogether lowers agent success significantly. These results show that the policy depends strongly on the belief input, while small perturbations to goal belief can diminish performance.

Table 6: Belief interventions on STAC after training completion. Values report the mean ± sample standard deviation over three seeds.

	Episode success (%)	
Belief condition	Training-support	Held-out configurations
Correct belief	69.25 ± 3.54	60.10 ± 1.23
Same-goal alternate history	69.11 ± 3.41	60.56 ± 0.90
Different-goal swap	64.04 ± 2.19	58.04 ± 0.95
Zero belief	17.01 ± 3.82	14.03 ± 3.05

F SUPPLEMENTARY ADAPTATION AND SEED-LEVEL RESULTS

Tables 7 and 8 provide the per-task results for the in-support and held-out benchmark comparisons in Fig. 2 (Sec. 5.1). Tables 9 and 10 report the per-task success rates corresponding to the structural and control-encoder comparisons in Figures 3a and 3b (Sec. 5.2).

Table 7: ML10 benchmark: for in-support configurations. Success percentages, mean ± sample standard deviation over three training seeds for each task. The highest average task success is denoted in bold. These results correspond to Figure 2.

Task	**VariBAD**	**AMAGO2**	**RL²-PPO**	**ECET**	**PEARL**	**STAC concat**	**STAC gating**
reach	37.7 ± 6.7	$\mathbf{48.3 \pm 21.5}$	29.0 ± 13.9	9.0 ± 10.1	33.3 ± 4.7	36.7 ± 11.6	26.7 ± 5.9
push	16.7 ± 20.8	32.3 ± 28.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	50.0 ± 25.2	$\mathbf{59.7 \pm 6.8}$
pick-place	33.7 ± 28.4	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	50.3 ± 8.3	$\mathbf{52.0 \pm 13.7}$
door-open	31.3 ± 54.3	0.0 ± 0.0	13.7 ± 23.7	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{91.0 \pm 4.4}$	85.0 ± 13.7
drawer-close	99.0 ± 1.0	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	98.7 ± 2.3	96.7 ± 1.2	$\mathbf{100.0 \pm 0}$	$\mathbf{100.0 \pm 0}$
button-press-topdown	98.3 ± 0.6	93.7 ± 11.0	60.0 ± 52.7	3.7 ± 6.4	0.0 ± 0.0	98.3 ± 2.9	$\mathbf{100.0 \pm 0}$
peg-insert-side	4.0 ± 6.9	1.3 ± 2.3	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	24.7 ± 12.4	$\mathbf{42.0 \pm 8.5}$
window-open	97.7 ± 1.2	$\mathbf{100.0 \pm 0.0}$	27.3 ± 47.3	39.3 ± 43.4	0.7 ± 0.6	$\mathbf{100.0 \pm 0.0}$	99.0 ± 1.7
sweep	60.7 ± 52.6	77.3 ± 21.4	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{95.3 \pm 7.2}$	93.0 ± 6.2
basketball	32.0 ± 31.5	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	43.0 ± 12.5	$\mathbf{52.0 \pm 9.2}$
Average	51.1 ± 7.5	45.3 ± 1.2	23.0 ± 9.0	15.1 ± 4.1	13.1 ± 0.4	68.9 ± 4.6	$\mathbf{70.9 \pm 3.5}$

Table 8: ML10 benchmark: for held-out configurations. Success percentages, mean ± sample standard deviation over three training seeds for each task. The highest average task success is denoted in bold. These results correspond to Figure 2.

Task	**VariBAD**	**AMAGO2**	**RL²-PPO**	**ECET**	**PEARL**	**STAC concat**	**STAC gating**
reach	0.0 ± 0.0	$\mathbf{34.3 \pm 9.0}$	21.0 ± 19.7	1.3 ± 2.3	23.7 ± 9.0	29.7 ± 6.0	18.3 ± 13.1
push	17.3 ± 25.8	26.3 ± 22.8	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	54.7 ± 11.1	$\mathbf{62.7 \pm 4.6}$
pick-place	2.3 ± 3.2	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	14.7 ± 9.0	$\mathbf{17.3 \pm 6.7}$
door-open	31.3 ± 53.4	0.0 ± 0.0	9.3 ± 16.2	0.0 ± 0.0	0.0 ± 0.0	93.0 ± 5.2	$\mathbf{95.0 \pm 6.2}$
drawer-close	99.0 ± 1.0	99.7 ± 0.6	$\mathbf{100.0 \pm 0.0}$	98.3 ± 2.9	97.7 ± 1.2	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$
button-press-topdown	97.7 ± 2.5	$\mathbf{98.3 \pm 2.9}$	45.3 ± 41.4	0.0 ± 0.0	0.0 ± 0.0	91.3 ± 6.0	98.0 ± 2.0
peg-insert-side	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{12.0 \pm 5.3}$	9.7 ± 6.8
window-open	78.3 ± 13.7	$\mathbf{99.3 \pm 1.2}$	29.0 ± 50.2	24.7 ± 34.4	0.0 ± 0.0	93.7 ± 8.5	89.7 ± 10.5
sweep	53.3 ± 47.3	83.3 ± 14.4	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	$\mathbf{85.0 \pm 6.6}$	78.3 ± 17.2
basketball	27.3 ± 26.1	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0	46.7 ± 6.7	53.3 ± 11.0
Average	40.7 ± 2.1	44.1 ± 1.6	20.5 ± 8.7	12.4 ± 3.5	12.1 ± 1.0	$\mathbf{62.1 \pm 0.7}$	$\mathbf{62.2 \pm 3.0}$

Table 9: Per-task success rates for the architecture ablations. Values report the mean ± sample standard deviation over three seeds. The highest average task success is denoted in bold. These results correspond to Figure 3a.

Task	**STAC concat**	**STAC gating**	**M-noIso**	**M-noRF**	**M-noSep**	**M-combo**
reach	36.0 ± 11.0	36.7 ± 11.6	$\mathbf{38.3 \pm 9.6}$	32.0 ± 7.2	14.3 ± 2.9	12.7 ± 3.1
push	34.7 ± 3.1	41.0 ± 7.0	$\mathbf{49.7 \pm 8.5}$	2.7 ± 2.9	0.3 ± 0.6	0.0 ± 0.0
pick-place	37.0 ± 13.9	$\mathbf{44.3 \pm 6.8}$	43.0 ± 14.1	0.3 ± 0.6	0.0 ± 0.0	0.0 ± 0.0
door-open	$\mathbf{95.0 \pm 4.0}$	92.3 ± 8.3	91.3 ± 4.2	84.3 ± 5.9	0.3 ± 0.6	0.0 ± 0.0
drawer-close	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	98.0 ± 3.5	79.7 ± 5.5	67.3 ± 12.0
button-press-topdown	95.7 ± 4.9	$\mathbf{99.0 \pm 1.0}$	96.7 ± 2.9	66.7 ± 34.4	1.0 ± 1.0	0.3 ± 0.6
peg-insert-side	17.3 ± 8.7	$\mathbf{33.7 \pm 7.2}$	16.3 ± 5.0	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
window-open	98.7 ± 2.3	$\mathbf{99.7 \pm 0.6}$	98.3 ± 2.9	96.3 ± 3.1	11.0 ± 2.0	14.0 ± 2.0
sweep	93.7 ± 1.2	$\mathbf{98.0 \pm 2.0}$	96.7 ± 2.1	4.0 ± 6.1	0.0 ± 0.0	0.3 ± 0.6
basketball	46.3 ± 12.9	$\mathbf{57.0 \pm 4.4}$	45.7 ± 14.3	0.0 ± 0.0	0.0 ± 0.0	0.0 ± 0.0
Average	65.4 ± 2.6	$\mathbf{70.2 \pm 2.3}$	67.6 ± 4.3	38.4 ± 4.3	10.7 ± 0.6	9.5 ± 1.2

Table 10: Per-task success rates for the control-encoder comparison under the reference configuration. Values report the mean ± sample standard deviation over three seeds. The highest average task success is denoted in bold. The Transformer column reuses the original STAC concat reference. These results correspond to Figure 3b.

Task	Transformer	GRU	Frame-stack MLP
reach	$\mathbf{36.0 \pm 11.0}$	25.0 ± 5.3	25.3 ± 2.3
push	$\mathbf{34.7 \pm 3.1}$	2.3 ± 2.3	1.7 ± 1.2
pick-place	$\mathbf{37.0 \pm 13.9}$	0.0 ± 0.0	0.0 ± 0.0
door-open	$\mathbf{95.0 \pm 4.0}$	88.0 ± 8.7	12.0 ± 3.6
drawer-close	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$	$\mathbf{100.0 \pm 0.0}$
button-press-topdown	$\mathbf{95.7 \pm 4.9}$	90.0 ± 1.7	28.0 ± 13.5
peg-insert-side	$\mathbf{17.3 \pm 8.7}$	0.0 ± 0.0	0.0 ± 0.0
window-open	98.7 ± 2.3	$\mathbf{99.0 \pm 1.0}$	86.7 ± 12.1
sweep	$\mathbf{93.7 \pm 1.2}$	0.3 ± 0.6	0.0 ± 0.0
basketball	$\mathbf{46.3 \pm 12.9}$	0.0 ± 0.0	0.0 ± 0.0
Average	$\mathbf{65.4 \pm 2.6}$	40.5 ± 0.2	25.4 ± 1.2