TETHER: TASK-EVOLVING TEACHER FOR HISTORY-EFFICIENT REGULARIZATION IN CONTINUAL ROBOT LEARNING

Anonymous authors
Paper under double-blind review

ABSTRACT

In continual robot manipulation, sequential fine-tuning can cause a language-conditioned flow-matching policy to generate similar actions under distinct past task conditions. We introduce TETHER (**T**ask-**E**volving **T**eacher for **H**istory-**E**fficient **R**egularization), which preserves past condition–action correspondences in a shared policy without storing individual demonstrations. The preceding policy serves as a frozen, task-evolving teacher. Task Statistics Memory (TSM) retains compact state-cluster statistics and local state-to-visual-feature maps to reconstruct paired proprioceptive and visual conditions. The teacher estimates an action coordinate for each reconstructed condition, defining action queries at which the student matches its velocity while learning new tasks through flow matching. Magnitude-Aware Regularization (MAR) gives small teacher pose motions greater relative weight, strengthening directional guidance that popular squared-error matching can underemphasize. We evaluate TETHER on simulated benchmarks and real-robot tasks. TETHER achieves strong performance compared to the state-of-the-art methods, notably with the best Area Under the Curve (AUC) and lowest Negative Backward Transfer (NBT) across various tasks, showing that its regularization effectively mitigates forgetting of previously learned condition–action correspondences.

1 INTRODUCTION

Robots in open-ended environments must acquire new manipulation skills without losing the ability to follow earlier instructions, a challenge known as continual or lifelong learning (Parisi et al., 2019). Continual robot learning therefore requires retaining procedural knowledge, as similar visual scenes may require different actions depending on the task goal (Liu et al., 2023). Recently, vision-language-action (VLA) models have shown that architectures combining visual perception, language understanding, and action generation can tackle a wide range of robotic tasks (Römer et al., 2026; Black et al., 2025). Diffusion- or flow-based policies generate diverse action chunks

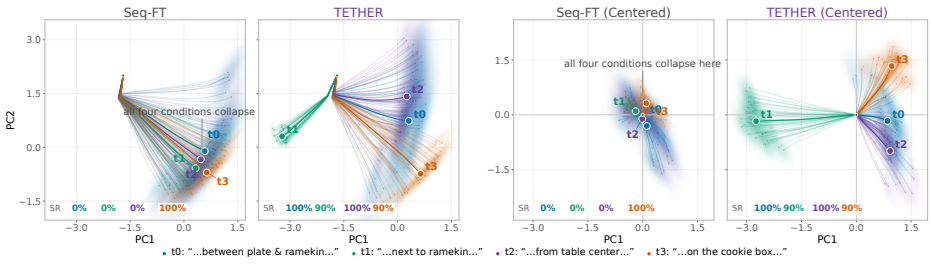


Figure 1: **Condition-specific action generation** Pose-action denoising paths after learning tasks 0–3 sequentially, using shared initial noise and 100 Euler steps, projected onto two principal components. Colors identify task conditions. Left: original paths. Right: after removing the across-condition mean. Thick paths and markers show condition means and endpoints, and SR denotes requested-task success rate.

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.

conditioned on observations, robot states, and language instructions (Chi et al., 2023; Lipman et al., 2023). However, naively training on the newest demonstrations is prone to catastrophic forgetting (French, 1999), potentially producing similar action trajectories under distinct earlier task conditions, as illustrated for sequential fine-tuning (Seq-FT) in Figure 1.

Prior work protects different substrates of a learned policy. Weight regularization and output distillation limit parameter or prediction drift (Kirkpatrick et al., 2017; Zenke et al., 2017; Li & Hoiem, 2018; Roy et al., 2025), while pruning, adapters, and hierarchical skill libraries allocate task-specific capacity (Mallya & Lazebnik, 2018; Liu et al., 2024; Wan et al., 2024). Module selection often requires task identity, whereas CLARE (Römer et al., 2026) removes this oracle through autonomous adapter routing, while retaining an expanding set of adapters as the number of tasks increases.

Experience Replay (ER) mitigates forgetting by revisiting stored condition–action examples (Lopez-Paz & Ranzato, 2017; Rolnick et al., 2019; Wan et al., 2024) in a replay buffer. Generative replay instead uses a learned generator (Gao et al., 2021), while MLR-IFA stores compressed multimodal latents and promotes task-separated representations (Yu et al., 2026). These methods reduce raw-trajectory storage but still require a generator or replay memory whose memory increases with task numbers. When demonstrations cannot be retained due to storage or privacy constraints, the challenge is to reconstruct compatible prior conditions.

For a flow-matching action expert, the aforementioned conditions jointly parameterize a time-dependent velocity field that transports samples from an initial noise distribution toward valid action trajectories. Integrating this field over time produces the final action chunk (Lipman et al., 2023; Black et al., 2025; Römer et al., 2026). Matching parameters or distilling only on new-task inputs does not directly protect its response under old conditions. OMP (Fang et al., 2026) shows that directional alignment matters for accurate one-step MeanFlow action generation, but does not address forgetting issues in lifelong learning settings.

In this paper, we propose TETHER, which uses the previous policy as a frozen, task-evolving teacher to guide the current action policy. Task Statistics Memory (TSM) retains state-cluster statistics and local state-to-feature maps to regenerate compatible past condition pairs without storing individual samples. Given a reconstructed pair and its retained instruction, the teacher predicts an action under the corresponding prior-task condition. During flow-matching training on current demonstrations, the student velocity is regularized toward the teacher's velocity, preserving previously learned condition–action correspondences within a shared policy. Magnitude-Aware Regularization (MAR) emphasizes small pose motions to preserve the teacher's directional guidance across tasks.

Contributions. Our main contributions are as follows:

- ***Task-Evolving Teacher for Regularizing Action Generation:*** TETHER introduces a task-evolving teacher as a regularizer for flow-matching-based action generation in VLA backbone (Römer et al., 2026; Black et al., 2025), preserving past condition–action correspondences and mitigating overfitting to the latest task in continual learning settings.
- ***Task Statistics Memory (TSM):*** In TETHER, instead of storing whole trajectories as in the experience replay (Rolnick et al., 2019; Chaudhry et al., 2019), we construct memory-efficient statistics to recover the action generation conditions of previous tasks, including observations, proprioception, and gripper status, to feasibly generate regularization by a task-evolving teacher.
- ***Magnitude-Aware Regularization (MAR):*** In robot continual learning, it is critical for an action expert to preserve the direction of task-specific expert actions conditioned on task-specific contexts. Otherwise, the generated action in a given scene may remain largely unchanged across different task conditions, as illustrated in Figure 1. To address this issue, TETHER proposes Magnitude-Aware Regularization (MAR), which preserves the directional signal of teacher guidance using task conditions regenerated through TSM.

2 PRELIMINARIES

2.1 PROBLEM DEFINITION

In this paper, we consider a stream of robot manipulation tasks $\{\mathfrak{T}_j\}_{j\geq 1}$. A task $\mathfrak{T}_j = (\rho_j^0, \ell_j, g_j)$ specifies an initial distribution ρ_j^0 over robot and environment states, a language instruction ℓ_j, and

a success criterion g_j for the resulting rollout. Tasks arrive sequentially, and future tasks need not be known in advance (Römer et al., 2026). At control step t, the raw observation comprises camera inputs $I_t = (I_t^1, \ldots, I_t^C)$ from C views, robot proprioception $s_t \in \mathbb{R}^{d_s}$ (including gripper state), and the instruction ℓ_j. Here d_s and d_o are the corresponding representation dimensions. The camera and proprioceptive inputs may include a short observation history. We distinguish the images I_t from their encoded features o_t obtained by the visual encoder $f_\phi(\cdot)$, i.e., $o_t = f_\phi(I_t) \in \mathbb{R}^{d_o}$, and write the policy condition for the j-th task as $c_t^j = (s_t^j, o_t^j, \ell_j)$.

For each task $\mathfrak{T}_j$, we have expert demonstration $\mathcal{D}_j = \{(c_t^j, a_t^j)\}_{t=1}^{T_j}$, where T_j denotes the number of control steps in the j-th demonstration; $u_t \in \mathbb{R}^A$ is an expert control with an action dimension A; and $a_t = (u_t, \ldots, u_{t+H-1}) \in \mathbb{R}^{H\times A}$ with an H-step action chunk (Chi et al., 2023; Black et al., 2025). At a new learning stage of the n-th task, for simplicity, we let the current task index as $\mathcal{T}_{\text{new}} = n$ and the previous task indices as $\mathcal{T}_{\text{past}} = \{1, 2, \cdots, n-1\}$ and define $\mathcal{D}_{\text{new}} = \mathcal{D}_n$ and $\mathcal{D}_{\text{past}} = \{\mathcal{D}_j\}_{j\le n-1}$. In the continual robot learning setting, the demonstration datasets from earlier stages, i.e., $\mathcal{D}_{\text{past}}$, are not available at the n-th stage unless an additional memory buffer is constructed to retain them.

2.2 Conditional flow-matching policy

To learn from high-dimensional, multi-modal demonstrations, we train the action policy with flow matching (Lipman et al., 2023), and then policy predicts a velocity $v_\theta(x_t, t \mid c) \in \mathbb{R}^{H\times A}$ to generate the target action a. From independent standard Gaussian noise $\varepsilon \in \mathbb{R}^{H\times A}$ of the same shape, with $\varepsilon \sim \mathcal{N}(0, I_{HA})$, and the interpolation step $t \sim \mathcal{U}(0, 1)$, the interpolated action x_t and target velocity $v^\star$ can be written as

$$x_t = (1-t)\varepsilon + ta, \qquad v^\star = a - \varepsilon. \tag{1}$$

We train the conditional action policy on new demonstrations $\mathcal{D}_{new}$ with

$$\mathcal{L}_{\text{FM}}(\theta) = \mathbb{E}_{(c,a)\sim\mathcal{D}_{\text{new}},\varepsilon\sim\mathcal{N}(0,I_{HA}),t\sim\mathcal{U}(0,1)}\left[\left(v_\theta(x_t, t \mid c) - v^\star\right)^2\right]. \tag{2}$$

Equation (2) trains the policy to predict the flow velocity from Gaussian noise toward the demonstrated action conditioned on c. At inference, we sample $\varepsilon \sim \mathcal{N}(0, I_{HA})$ and integrate $\dot{x}_t = v_\theta(x_t, t \mid c)$ from $t = 0$ to 1 to obtain the action chunk. In continual learning, this objective learns the new task, while the regularization introduced below preserves the velocity guidance from past tasks. Appendix A presents related work and discussions.

3 Methodology

To address latest-task bias, TETHER preserves action generation conditioned on task-specific context while continuing to learn new tasks via task-evolving teacher construction. Figure 2 summarizes the overall TETHER training procedure with teacher regularization: new demonstrations supervise the student through the flow-matching loss, augmented by teacher-velocity regularization, while the Task Statistics Memory (TSM) reconstructs past conditions for teacher–student velocity matching. Additionally, Magnitude-Aware Regularization (MAR) assigns greater relative weight to pose-component matching errors when the teacher's pose-motion magnitude is small, whereas a standard MSE loss may underweight important directional velocity guidance in such case.

3.1 Constructing a task-evolving teacher for regularization

In TETHER, we freeze the visual and language encoders, which, together with the robot state s, define the task condition c. The student policy to generate velocity field, denoted by v_θ, is trained at the current continual-learning stage and then becomes the frozen teacher, denoted by v_T, for the next stage. We defer implementation details to Appendix J. At the state n, to feasibly utilize the velocity guidance through $v_T(\cdot|c_j)$ for $j \in \mathcal{T}_{\text{past}}$, we need past task conditions $c_{j<n}$. Instead of storing corresponding trajectories of $\mathcal{D}_{\text{past}}$, we construct the anchor-query distribution $\mathcal{Q}_j$ to generate such $c_{j<n}$ via TSM $\mathcal{M}_j$ for each $j \in \mathcal{T}_{\text{past}}$ with the fixed instruction ℓ_j.

Paired state–feature reconstruction. In reality, visual observations reflect the robot's configuration, motivating visual-feature reconstruction conditioned on robot state rather than independent

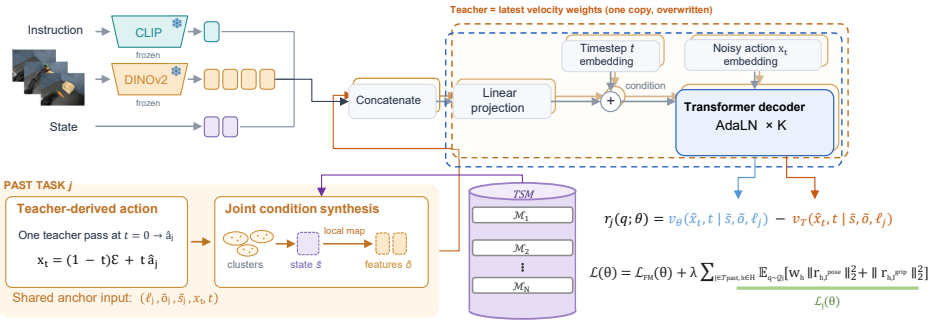


Figure 2: **Overview of TETHER.** New demonstrations train the student through flow matching. For past tasks, stored TSM statistics and instructions reconstruct prior conditions, while the frozen teacher provides action coordinates and velocity targets. The student and teacher are evaluated on the same reconstructed queries, with MAR assigning greater relative weight to small teacher pose motions during velocity matching. After training, the student becomes the teacher for the next stage.

modeling for the distribution of visual observations. Accordingly, we factor the joint distribution as $\hat{p}_j(s, o) = \hat{p}_j(s)\hat{p}_j(o \mid s)$. The low-dimensional state representation ($d_s = 16$ versus $d_o = 3072$ for visual features) provides a compact space for density estimation and grouping similar context features. Within each group, local maps approximate the conditional mean of the visual features. Conditioning reconstructed features on the sampled state preserves the modeled state–feature relationship and reduces implausible combinations that could arise from independent sampling. All statistics and local maps are estimated while the demonstrations for task j are available and remain fixed thereafter.

We cluster standardized demonstration states using k-means clustering (MacQueen, 1967), and Appendix J.13 provides its implementation details. For each of the retained clusters, $k \in \{1, 2, \cdots, K_j\}$ at the j-th task, we store its frame fraction as $\pi_{j,k} = N_{j,k}/N_j$, i.e., the number of frames assigned to cluster k ($N_{j,k}$) divided by the total number of frames in task j (N_j), raw-state mean $\mu^s_{j,k}$ and covariance $\Sigma_{j,k}$, and mean visual feature $\mu^o_{j,k}$. Given task-wise state mean $\bar{s}_j$ and standard deviation $\bar{\sigma}_j$ for all states s_j in $\mathcal{D}_j$, we define normalization function f_z to compute normalized state z_j via $z_j = f_z(s_j; \mu^s_j, \Sigma_j)$ whose elements are computed by $z_{j,d} = (s_{j,d} - \bar{s}_{j,d})/\bar{\sigma}_{j,d}$ for $d \in \{1, \ldots, d_s\}$.

The normalized state mean for the k-th cluster, i.e., $\bar{z}_{j,k}$, is easily recoverable with $\mu^s_{j,k}, \Sigma_{j,k}$. Instead of an explicit analytical state-to-visual feature mapping, in TETHER, we approximate each cluster's conditional visual feature mean by $\mu^o_{j,k} + A_{j,k}(f_z(s_{j,k}) - \bar{z}_{j,k})$. Here, we estimate $A_{j,k} \in \mathbb{R}^{d_o \times d_s}$ by ridge regression (Hoerl & Kennard, 1970), which admits a closed-form solution and regularizes the fit with limited cluster samples.

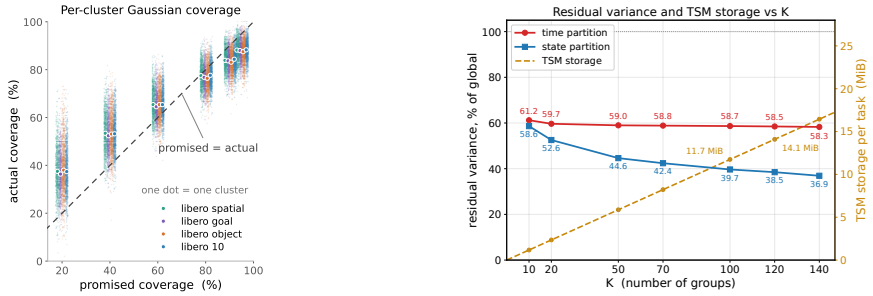
Finally, TSM $\mathcal{M}_j$ for the j-th stores the following statistics:

$$\mathcal{M}_j = \Big((\bar{s}_j, \bar{\sigma}_j),\ \bar{\rho}^2_j,\ \big\{ \big(\pi_{j,k},\ (\mu^s_{j,k}, \Sigma_{j,k}),\ (\mu^o_{j,k}, A_{j,k})\big) \big\}_{k=1}^{K_j} \Big). \tag{3}$$

The scalar $\bar{\rho}^2_j$ provides the pose-action magnitude reference for anchor weighting (Section 3.2). Besides $\mathcal{M}_j$, we store the task instruction ℓ_j separately. With $\pi_j = (\pi_{j,1}, \ldots, \pi_{j,K_j})$, we reconstruct a pair $(\tilde{s}_j, \tilde{o}_j)$ by:

$$\tilde{s}_j \sim \mathcal{N}(\mu^s_{j,k}, \Sigma_{j,k}), \qquad \tilde{o}_j = \mu^o_{j,k} + A_{j,k}\big(z_j(\tilde{s}_j) - \bar{z}_{j,k}\big). \tag{4}$$

where k is sampled from Categorical distribution, i.e., $k \sim \mathrm{Cat}(\pi_j)$. We denote the mapping from $\tilde{s}_j$ to $\tilde{o}_j$ as a local map. The sampler in Equation (4) combines the cluster prior, $\hat{p}_j(k) = \pi_{j,k}$, the state model, $\hat{p}_j(s \mid k) = \mathcal{N}(s; \mu^s_{j,k}, \Sigma_{j,k})$, and the conditional feature model, $\hat{p}_j(o \mid s, k)$, represented as a point mass at the prediction of the local map. Bayes' rule gives $\hat{p}_j(k \mid s) \propto \pi_{j,k} \cdot \mathcal{N}(s; \mu^s_{j,k}, \Sigma_{j,k})$, yielding $\hat{p}_j(o \mid s) = \sum_k \hat{p}_j(k \mid s)\hat{p}_j(o \mid s, k)$. Since an explicit analytical mapping from robot states to visual features is unavailable, we estimate the visual feature associated with each sampled state using this cluster-specific ridge-regression map. Frozen encoders keep the visual-feature representation consistent across stages.



(a) (b)

Figure 3: **State-based condition reconstruction.** (a) Expected versus observed coverage of Mahalanobis-distance regions at several reference points (x-axis) of $\{20, 40, 60, 80, 90, 95\}\%$. Dots are clusters of the normalized state z_j for all $j \in \mathcal{T}$ in LIBERO suites, large markers are suite medians, and the diagonal denotes agreement. (b) Within-group visual feature variance and TSM storage versus group count.

Figure 3 examines the Gaussian state model and state-based feature grouping. Figure 3 (a) compares Gaussian-predicted coverage with the fraction of observed states inside the corresponding Mahalanobis-distance regions. At the suite-median level, regions with high predicted coverage contain most observed states, supporting the validity of the proposed Gaussian sampling. Empirical states are more concentrated in central regions and have heavier outer tails than the Gaussian reference. Figure 3 (b) measures within-group visual-feature variance normalized by the variance of total visual features. For comparison, we also evaluate the variance residual obtained by partitioning trajectories into uniform time bins. With 50 episodes and $K = 100$, state grouping leaves a residual feature variance of 39.7%, compared with 58.7% for uniform time-bin partitioning. State grouping therefore explains more visual-feature variation than uniform time-bin grouping, supporting the validity of state-conditioned feature reconstruction. Appendix F provides the factorization and estimators, and Appendices J.8 and G provide details of diagnostics and sensitivity analysis according to the TSM cluster count K, respectively.

Action coordinates under the same condition. To construct the velocity-matching query q, we first estimate an action endpoint under $(\tilde{s}_j, \tilde{o}_j, \ell_j)$ by taking a single *forward-Euler step* of the teacher v_T along the ODE from $t = 0$ to $t = 1$ with unit step size. We then interpolate this endpoint with independent anchor noise ε to obtain $\hat{x}$:

$$\hat{a}_j = x_0 + v_T(x_0, t = 0 \mid \tilde{s}_j, \tilde{o}_j, \ell_j), \qquad \hat{x}_t = (1 - t)\varepsilon + t\hat{a}_j. \tag{5}$$

Here, x_0 and ε are independent draws from $\mathcal{N}(0, I_{HA})$ and $t \sim \mathcal{U}(0, 1)$. The reconstructed coordinate $\hat{x}_t$ differs from the demonstration-based x_t in Equation (1): x_0 serves as the teacher's initial input, whereas ε is used to construct the noisy anchor. Under the ideal conditional-mean flow field at $t = 0$, the resulting endpoint corresponds to $\mathbb{E}[a \mid c]$; with the learned teacher, we use this endpoint as an approximation. The teacher and student are then matched at the same query $(\hat{x}_t, t \mid \tilde{s}_j, \tilde{o}_j, \ell_j)$, avoiding the need to reproduce the teacher's full ODE trajectory. Equations (4) and (5) define $\mathcal{Q}_j$, the joint distribution over the complete anchor query $q = (\tilde{s}_j, \tilde{o}_j, \hat{x}_t, t, \varepsilon)$ constructed from $\mathcal{M}_j$, v_T, and ℓ_j. Its state–feature marginal is denoted by $\hat{p}_j(s, o)$, while the teacher v_T provides the conditional action endpoint $\hat{a}_j$, referred to as the *action coordinate* in this paper, which is used to construct the query.

3.2 Magnitude-Aware Regularization

In TETHER, MAR gives greater relative weight to queries with smaller teacher pose-motion magnitudes during student–teacher velocity matching, helping preserve directional guidance that may otherwise be underweighted by the standard MSE loss (settings: Appendix J.14). For each past task $j \in \mathcal{T}_{\text{past}}$, we define the shared-input residual r_j and the corresponding loss $\mathcal{L}_j^{\text{base}}(\theta)$ as:

$$r_j(q; \theta) = v_\theta(\hat{x}_t, t \mid \tilde{s}_j, \tilde{o}_j, \ell_j) - v_T(\hat{x}_t, t \mid \tilde{s}_j, \tilde{o}_j, \ell_j). \tag{6}$$

$$\mathcal{L}_j^{\text{base}}(\theta) = \mathbb{E}_{q \sim \mathcal{Q}_j}\left[r_j(q; \theta)^2\right]. \tag{7}$$

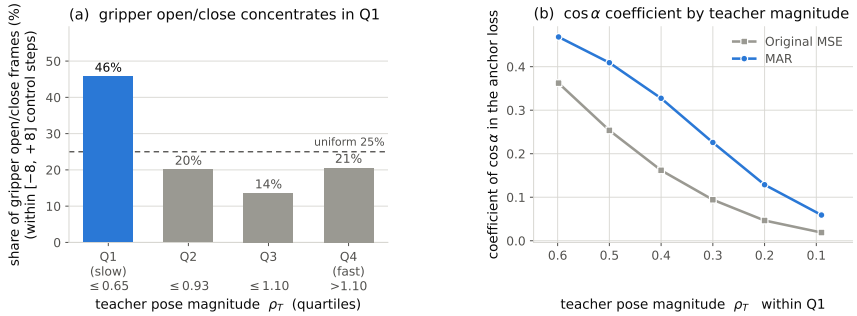


Figure 4: **Magnitude-Aware Regularization (MAR).** Measurements use LIBERO-Spatial demonstration data. (a) The lowest teacher pose-magnitude quartile (Q1) contains the largest share of frames near gripper transitions. (b) Within Q1, the angular-alignment coefficient of the anchor loss, $\rho_{S,h}\rho_{T,h}$ without MAR and $w_h\rho_{S,h}\rho_{T,h}$ with MAR (common factor of two omitted); MAR raises the coefficient as the teacher pose magnitude decreases.

Magnitude measurement. Below, we omit the task index j for brevity and shared query arguments q when clear from context. The action index I ranges over $I_{\text{pose}} \cup I_{\text{grip}} = \{1, \ldots, A\}$, with $I_{\text{pose}} = \{1, \ldots, 6\}$ and $I_{\text{grip}} = \{7\}$ selecting the pose and gripper components, respectively. The subscript "pose" selects the components in I_{pose}. Recall that the flow-matching target is $v^\star = a - \varepsilon$ (Equation (1)). At the shared anchor query, define action proxies $a_S := v_\theta + \varepsilon$ and $a_T := v_T + \varepsilon$.

Their common noise can be canceled out as:

$$r = v_\theta - v_T = (a_S - \varepsilon) - (a_T - \varepsilon) = a_S - a_T. \tag{8}$$

At action step h, the pose magnitudes in normalized action coordinates are defined as:

$$\rho_{T,h} = \|a_{T,h,\text{pose}}\|_2, \qquad \rho_{S,h} = \|a_{S,h,\text{pose}}\|_2. \tag{9}$$

For nonzero pose proxies with angle α_h, the residual satisfies the following equations:

$$\|r_{h,\text{pose}}\|_2^2 = \rho_{S,h}^2 + \rho_{T,h}^2 - 2\rho_{S,h}\rho_{T,h}\cos\alpha_h, \tag{10}$$

$$\frac{\partial\|r_{h,\text{pose}}\|_2^2}{\partial\alpha_h} = 2\rho_{S,h}\rho_{T,h}\sin\alpha_h. \tag{11}$$

For fixed pose magnitudes, minimizing this loss encourages directional alignment. Under the standard MSE loss, however, the angular penalty is scaled by $2\rho_{S,h}\rho_{T,h}$; thus, for a fixed student magnitude and angular error, a smaller teacher magnitude weakens the penalty, consistent with the observation in (Fang et al., 2026). To address this effect, MAR increases the relative contribution of queries with small teacher responses.

Weights and interpretation. Rather than introducing an additional direction-preserving term (Fang et al., 2026; Wang et al., 2024), we weight the entire squared pose residual. Let $\bar{\rho}_{T,j}^2$ be the median of squared pose-action magnitudes $\|a_{T,h,\text{pose}}\|_2^2$ across task j's demonstrations and action steps, stored in $\mathcal{M}_j$. We compute inverse-magnitude weights and normalize them to unit batch mean:

$$\widetilde{w}_h = \frac{1}{\rho_{T,h}^2 + \bar{\rho}_j^2}, \qquad w_h = \frac{\widetilde{w}_h}{\frac{1}{N_{\text{mini}}}\sum_{b=1}^{N_{\text{mini}}}\widetilde{w}_h^{(b)}}. \tag{12}$$

where N_{mini} is the mini-batch size. Appendix J.3 gives the batch-wise construction. The positive reference $\bar{\rho}_j^2$ prevents inverse weights from diverging as teacher pose magnitude approaches zero and moderates weight contrast across responses. Applying this weight to Equation (10) gives

$$w_h\|r_{h,\text{pose}}\|_2^2 = w_h(\rho_{S,h}^2 + \rho_{T,h}^2) - 2w_h\rho_{S,h}\rho_{T,h}\cos\alpha_h. \tag{13}$$

For interpretation, let us consider a single mini-batch, i.e., $N_{\text{mini}} = 1$. Then, the cosine coefficient c_α now has magnitude as $c_\alpha = 2\rho_{S,h}\rho_{T,h}/(\rho_{T,h}^2 + \bar{\rho}_j^2)$. Without $\bar{\rho}_j$ term $c_\alpha \to \infty$ as $\rho_{T,h} \to 0$, which can cause instability during training. Thus, we consider $\bar{\rho}_j$. For small $\rho_{T,h}$, c_α magnifies with $1/\rho_{T,h}^2$ term. In the following empirical analysis in Figure 4, we evaluate how the proposed

MAR actually preserves the directional guidance compared to the original MSE. Preserving this directional guidance is particularly important in continual robot learning, as the policy can generate similar vector-field directions even with different task conditions, as illustrated in Figure 1. This argument is further supported by the strong AUC and NBT performance of TETHER in various experiments in Section 4. Appendix E derives and separates the relative-magnitude and directional contributions. We exclude the gripper from magnitude computation because its unit-magnitude encoding can mask differences among small pose motions, while retaining its matching error with unit weight. Weights remain fixed during student differentiation. Section 3.3 gives the combined training objective.

Empirical analysis. Figure 4(a) shows that the lowest teacher pose-magnitude quartile (Q1) contains the largest share of frames near gripper opening or closing, motivating directional preservation for small pose responses. Figure 4(b) compares the angular-alignment coefficients $\rho_{S,h}\rho_{T,h}$ without MAR and $w_h\rho_{S,h}\rho_{T,h}$ with MAR, omitting their common factor of two. Within a batch, w_h increases as teacher pose magnitude decreases. The larger MAR coefficients in the plotted Q1 range indicate greater relative emphasis on directional agreement for small teacher responses. Together with (a), this supports emphasizing the pose-magnitude range where near-transition frames concentrate. Appendix J.14 details the comparison.

3.3 Overall objective and algorithm

We combine current-task flow matching with past-task anchors, adopting MAR-weighted pose and gripper error. Then, the final loss function in TETHER is expressed as

$$\mathcal{L}(\theta) = \mathcal{L}_{\mathrm{FM}}(\theta) + \lambda \sum_{j\in\mathcal{T}_{\mathrm{past}},h\in H} \mathbb{E}_{q\sim\mathcal{Q}_j}\left[w_h \left\|r_{h,I_{\mathrm{pose}}}\right\|_2^2 + \left\|r_{h,I_{\mathrm{grip}}}\right\|_2^2\right]. \tag{14}$$

Here $r = v_\theta - v_T$ is evaluated at $(\hat{x}_t, t \mid \tilde{s}, \tilde{o}, \ell_j)$, and $\lambda > 0$ is the regularization coefficient. The expectation averages over queries q and action steps h.

Training procedure. At each update, the student learns new demonstrations and matches the frozen teacher at reconstructed inputs for every past task. After each stage, we store $\mathcal{M}_j$ and ℓ_j for each new task and replace the teacher with the updated student. Algorithm 1 in Appendix H provides the detailed procedure. In addition, *Task-Parallel Anchoring (TPA)* groups past-task queries for efficient execution (Appendix H.1). Inference requires only the learned policy and its ordinary condition inputs.

4 Experiments

In this section, we evaluate TETHER on both simulation-based benchmarks and customized real-robot tasks. For comparison, we consider several state-of-the-art (SOTA) methods, including LOTUS (Wan et al., 2024), MLR (Yu et al., 2026), and CLARE (Römer et al., 2026).

4.1 Comparative evaluation on LIBERO benchmarks

Setup. We evaluate LIBERO-Object, LIBERO-Goal, and LIBERO-50 under the LOTUS protocol (Wan et al., 2024), reporting forward transfer (FWT), negative backward transfer (NBT), and success-rate AUC. Higher FWT/AUC and lower NBT are better. Sequential fine-tuning, ER, and CLARE are our runs. Other baseline results are reported values from the cited sources. Appendix J.9 provides detailed settings.

Lifelong performance. TETHER attains the lowest listed mean NBT and highest listed mean AUC on all three suites (Table 1, with baseline references in Appendix J.4). It is the only entry with negative NBT: past tasks end above their success at acquisition. CLARE's near-zero NBT instead comes from parameter isolation (Römer et al., 2026), whereas TETHER obtains backward transfer with a single shared policy. The AUC gain over ER is largest where ER forgets most (LIBERO-50 and LIBERO-Goal, $+6.9$ and $+10.2$). Figure 1 and Appendix B show conditional responses and environment rollouts.

Table 1: **Lifelong results under the LOTUS protocol** on LIBERO-Object, LIBERO-50 and LIBERO-Goal (mean ± std over three seeds). †Run on our backbone under the same protocol; other baselines as reported by Yu et al. (2026) on their own backbones (BUDS and LOTUS from Wan et al. (2024)); NA: not reported. Best per column in bold.

	LIBERO-Object			LIBERO-50			LIBERO-Goal		
Method	AUC↑	NBT↓	FWT↑	AUC↑	NBT↓	FWT↑	AUC↑	NBT↓	FWT↑
BUDS	$47.0^{\pm 1.0}$	$21.0^{\pm 1.0}$	$52.0^{\pm 2.0}$	$33.0^{\pm 3.0}$	$50.0^{\pm 4.0}$	$29.0^{\pm 3.0}$	$42.0^{\pm 1.0}$	$39.0^{\pm 1.0}$	$50.0^{\pm 1.0}$
LOTUS	$65.0^{\pm 3.0}$	$11.0^{\pm 1.0}$	$74.0^{\pm 3.0}$	$45.0^{\pm 2.0}$	$43.0^{\pm 1.0}$	$39.0^{\pm 2.0}$	$56.0^{\pm 1.0}$	$30.0^{\pm 1.0}$	$61.0^{\pm 3.0}$
IsCiL	$66.3^{\pm 3.7}$	$11.9^{\pm 5.1}$	$71.7^{\pm 1.9}$	$37.7^{\pm 2.1}$	$15.0^{\pm 4.8}$	$47.8^{\pm 1.3}$	$60.5^{\pm 2.5}$	$19.4^{\pm 4.1}$	$70.4^{\pm 3.6}$
M2Distill	$69.0^{\pm 4.0}$	$8.0^{\pm 5.0}$	$75.0^{\pm 3.0}$	NA	NA	NA	$57.0^{\pm 2.0}$	$20.0^{\pm 3.0}$	$71.0^{\pm 1.0}$
TAIL	$38.2^{\pm 2.5}$	$22.5^{\pm 2.1}$	$41.5^{\pm 2.9}$	$20.1^{\pm 3.3}$	$19.3^{\pm 9.4}$	$31.8^{\pm 3.0}$	$37.4^{\pm 1.3}$	$22.7^{\pm 1.1}$	$41.0^{\pm 1.2}$
MLR	$77.6^{\pm 3.0}$	$12.3^{\pm 2.4}$	$83.3^{\pm 2.6}$	$54.7^{\pm 2.4}$	$20.6^{\pm 9.0}$	$58.0^{\pm 4.0}$	$74.6^{\pm 2.7}$	$10.0^{\pm 7.5}$	$78.8^{\pm 2.2}$
MLR + IFA	$79.4^{\pm 1.5}$	$11.4^{\pm 5.6}$	$84.6^{\pm 1.9}$	$56.1^{\pm 1.8}$	$8.6^{\pm 6.2}$	$60.8^{\pm 2.8}$	$77.2^{\pm 1.8}$	$6.9^{\pm 0.9}$	$80.0^{\pm 2.5}$
Sequential†	$56.6^{\pm 1.1}$	$79.6^{\pm 3.3}$	$\mathbf{93.7}^{\pm 1.3}$	$34.9^{\pm 1.2}$	$75.6^{\pm 1.9}$	$\mathbf{75.5}^{\pm 0.6}$	$43.0^{\pm 0.5}$	$83.3^{\pm 3.3}$	$\mathbf{82.5}^{\pm 1.2}$
ER†	$84.8^{\pm 3.3}$	$13.4^{\pm 10.6}$	$91.2^{\pm 2.2}$	$56.6^{\pm 2.0}$	$31.5^{\pm 1.1}$	$73.3^{\pm 2.1}$	$67.4^{\pm 3.6}$	$27.6^{\pm 7.7}$	$80.8^{\pm 0.7}$
CLARE†	$75.6^{\pm 1.8}$	$0.7^{\pm 2.0}$	$75.8^{\pm 2.6}$	$36.5^{\pm 1.0}$	$1.2^{\pm 2.6}$	$37.5^{\pm 1.9}$	$57.1^{\pm 2.9}$	$0.6^{\pm 4.5}$	$57.9^{\pm 5.2}$
TETHER	$\mathbf{85.3}^{\pm 2.1}$	$\mathbf{-3.8}^{\pm 2.4}$	$83.3^{\pm 3.1}$	$\mathbf{63.5}^{\pm 1.0}$	$\mathbf{-0.9}^{\pm 2.6}$	$62.9^{\pm 2.3}$	$\mathbf{77.6}^{\pm 1.1}$	$\mathbf{-3.6}^{\pm 2.4}$	$75.8^{\pm 5.9}$

Transfer to LIBERO-40. In this experiment, we consider a large number of continual tasks, where 40 continual tasks are evaluated after base-phase training. Under the CLARE protocol (Römer et al., 2026), TETHER improves AUC over ER (74.9 versus 60.2) and reduces NBT (-0.8 versus 31.3), while FWT is lower (74.4 versus 88.6). Figure 5 shows where this comes from: ER decays below the diagonal as later stages overwrite earlier tasks, whereas TETHER keeps past-task columns near their acquisition level at the cost of lower success on the diagonal.

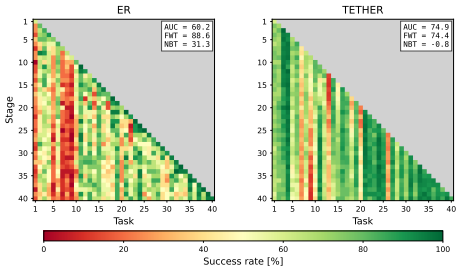
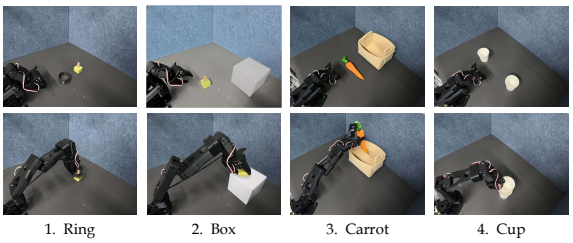


Figure 5: **Success-rate matrices on LIBERO-40.**

4.2 COMPARATIVE EVALUATION ON REAL-WORLD ROBOTS

Real robot tasks. For real-world robot tasks, we design four different tasks, such as Across Ring, Box, Carrot, and Cup (Figure 6(a)). Please refer to Appendix J.16 for the details of task descriptions. In Figure 6(b), our final policy succeeds in 33/40 rollouts versus our ER run's 26/40 (82.5% versus 65.0%). With ten trials per task and stage, TETHER records more final-stage successes on Box (6 versus 4) and the newest task, Cup (9 versus 4), while Ring and Carrot tie.



(a)

	1. RING		2. BOX		3. CARROT		4. CUP	
Stage	①	②	①	②	①	②	①	②
1	**10**	**10**	–	–	–	–	–	–
2	**10**	**10**	**8**	6	–	–	–	–
3	**10**	9	**6**	4	6	**7**	–	–
4	**10**	**10**	**6**	4	**8**	**8**	**9**	4

(b)

Figure 6: **Continual learning on four real robot tasks.** (a) Task sequence (details: Appendix J.16). (b) Successful rollouts out of ten per task and stage: ① TETHER, ② ER. TETHER has lower NBT (0.0 versus 4.4), with higher AUC (81.7 versus 64.8) and FWT (82.5 versus 67.5).

4.3 Ablation studies

Condition reconstruction and MAR. Independent Gaussian sampling gives 84.7 FWT but 72.2 NBT and only 33.8 AUC, indicating severe forgetting (Table 2). Four observation–state pairs sampled from data yield 83.9 AUC when shuffled and 84.8 when kept intact (Coupled). Full TETHER, including MAR (Section 3.2), reaches 90.2 AUC and 91.6 FWT: gains of 5.4 and 8.6 points over Coupled. Coupled has lower NBT (-3.2 versus -1.3).

Table 2: **Anchoring ablations.** Mean ± SD (three seeds).

Anchor inputs	FWT↑	NBT↓	AUC↑
Indep. Gaussian	84.7±2.4	+72.2±2.3	33.8±1.4
Shuffled	83.5±1.4	−1.1±1.6	83.9±0.7
Coupled	83.0±1.9	**−3.2±2.0**	84.8±0.4
TETHER	**91.6±1.5**	-1.3±0.7	**90.2±1.2**

4.4 Memory efficiency and efficient training

Storage efficiency. TSM uses 507 MB on LIBERO-50, reducing task-memory storage by 79.0% relative to our ER run and 64.8% relative to reported MLR, with higher AUC (63.5 versus 56.6 and 54.7, Table 3). Figure 7 separately includes the additional teacher snapshot.

Table 3: **Task memory and LIBERO-50 results.**

Method	Storage	AUC↑	NBT↓	FWT↑
ER	2,410 MB	56.6 ± 2.0	31.5 ± 1.1	73.3 ± 2.1
MLR	1,442 MB	54.7 ± 2.4	20.6 ± 9.0	58.0 ± 4.0
TETHER (ours)	507 MB	63.5 ± 1.0	-0.9 ± 2.6	62.9 ± 2.3

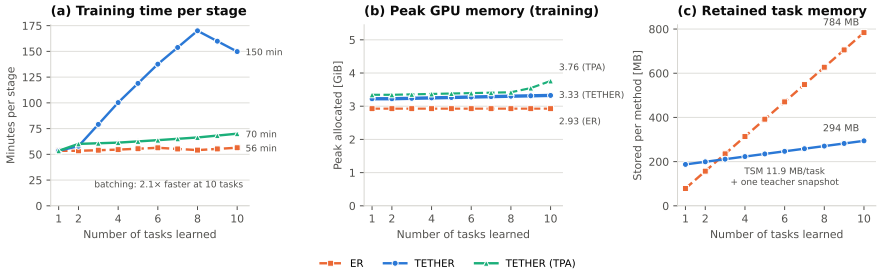


Figure 7: **Training cost and retained storage.** (a) Stage training time. (b) Peak allocated GPU memory. (c) Storage excluding the student: both implementations retain 11.9 MB of TSM per task and one fixed-size 175.5 MB teacher. TPA algorithm: Appendix H.1.

Training cost and retained storage. In our ten-task hardware comparison, TETHER reduces retained storage by 62.5% relative to ER (294 versus 784 MB, including one fixed-size teacher). TPA makes this practical, cutting stage time from 150 to 70 minutes (2.1×; ER: 56 minutes) while preserving task-wise anchors and reductions (Appendix H.1). Peak GPU memory is 3.76 GiB (ER: 2.93 GiB, Figure 7), and inference uses neither teacher nor TSM.

5 Conclusion

We presented TETHER, a continual imitation learning framework that preserves past condition–action correspondences without retaining individual demonstrations. TSM reconstructs paired states and visual features from compact task statistics, while teacher-derived action coordinates and MAR support velocity anchoring under past conditions. Under the LOTUS protocol, TETHER achieves the lowest listed mean NBT and highest listed mean AUC on LIBERO-Object, LIBERO-50, and LIBERO-Goal, while reducing retained task memory on LIBERO-50, implying strong retention capability of the learned policy. Future work could explore richer state–feature models for more accurate reconstruction of past conditions, and extend evaluation to longer task sequences and more diverse real-world environments.

AI USE STATEMENT

We used AI-assisted tools to support figure preparation and manuscript editing. For figures, these tools assisted with visual presentation and refinement. For the manuscript, they helped improve sentence structure, wording, and readability, refine the flow of explanations, and identify and correct grammatical and typographical errors. These uses aimed to communicate the methods and findings more clearly and consistently.

REPRODUCIBILITY STATEMENT

Appendix J.1 specifies the backbone, lifelong protocols, optimizer and training settings, anchor-query construction, weight post-processing, loss reduction, and TSM cluster and local-map settings used in all reported runs. Appendix F derives the local state-to-feature maps, Appendix E derives the MAR identity, and Algorithms 1 and 2 give the task-wise and TPA training procedures. Appendix J.4 identifies which baseline results are our runs and which are reported values, and Appendix J.16 describes the real-robot platform, data, and evaluation.

REFERENCES

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Robert Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, Szymon Jakubczak, Tim Jones, Liyiming Ke, Sergey Levine, Adrian Li-Bell, Mohith Mothukuri, Suraj Nair, Karl Pertsch, Lucy Xiaoyang Shi, Laura Smith, James Tanner, Quan Vuong, Anna Walling, Haohuan Wang, and Ury Zhilinsky. π_0: A vision-language-action flow model for general robot control. In *Proceedings of Robotics: Science and Systems (RSS XXI)*, Los Angeles, CA, USA, June 2025. doi: 10.15607/RSS.2025.XXI.010. URL https://arxiv.org/abs/2410.24164. arXiv:2410.24164, October 2024.

Pietro Buzzega, Matteo Boschini, Angelo Porrello, Davide Abati, and Simone Calderara. Dark experience for general continual learning: a strong, simple baseline. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 33, pp. 15920–15930, 2020. URL https://proceedings.neurips.cc/paper/2020/hash/b704ea2c39778f07c617f6b7ce480e9e-Abstract.html.

Arslan Chaudhry, Marcus Rohrbach, Mohamed Elhoseiny, Thalaiyasingam Ajanthan, Puneet K. Dokania, Philip H. S. Torr, and Marc'Aurelio Ranzato. On tiny episodic memories in continual learning. arXiv preprint arXiv:1902.10486, 2019. URL https://arxiv.org/abs/1902.10486.

Cheng Chi, Siyuan Feng, Yilun Du, Zhenjia Xu, Eric Cousineau, Benjamin Burchfiel, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. In *Robotics: Science and Systems (RSS)*, 2023.

Han Fang, Yize Huang, Yuheng Zhao, Paul Weng, Xiao Li, and Yutong Ban. OMP: One-step Meanflow policy with directional alignment. arXiv preprint arXiv:2512.19347, 2026. URL https://arxiv.org/abs/2512.19347. Accepted as a poster at the International Conference on Machine Learning (ICML) 2026; proceedings volume not yet published.

Robert M. French. Catastrophic forgetting in connectionist networks. *Trends in Cognitive Sciences*, 3(4):128–135, 1999. doi: 10.1016/S1364-6613(99)01294-2.

Chongkai Gao, Haichuan Gao, Shangqi Guo, Tianren Zhang, and Feng Chen. CRIL: Continual robot imitation learning via generative and prediction model. In *2021 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 6747–6754. IEEE, 2021. doi: 10.1109/IROS51168.2021.9636069. URL https://arxiv.org/abs/2106.09422.

Qiankun Gao, Chen Zhao, Bernard Ghanem, and Jian Zhang. R-DFCIL: Relation-guided representation learning for data-free class incremental learning. In *Computer Vision – ECCV 2022*, volume 13683 of *Lecture Notes in Computer Science*, pp. 423–439. Springer, 2022. doi: 10.1007/978-3-031-20050-2_25. URL https://arxiv.org/abs/2203.13104.

Manish Kumar Govind, Dominick Reilly, Smit Patel, Hieu Le, and Srijan Das. World action models enable continual imitation learning with recurrent generative replays. arXiv preprint arXiv:2606.27374, 2026. URL https://arxiv.org/abs/2606.27374.

Arthur E. Hoerl and Robert W. Kennard. Ridge regression: Biased estimation for nonorthogonal problems. *Technometrics*, 12(1):55–67, 1970. doi: 10.1080/00401706.1970.10488634.

Saihui Hou, Xinyu Pan, Chen Change Loy, Zilei Wang, and Dahua Lin. Learning a unified classifier incrementally via rebalancing. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 831–839, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Hou_Learning_a_Unified_Classifier_Incrementally_via_Rebalancing_CVPR_2019_paper.html.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A. Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, Demis Hassabis, Claudia Clopath, Dharshan Kumaran, and Raia Hadsell. Overcoming catastrophic forgetting in neural networks. *Proceedings of the National Academy of Sciences*, 114(13):3521–3526, 2017. doi: 10.1073/pnas.1611835114. URL https://arxiv.org/abs/1612.00796.

Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(12):2935–2947, 2018. doi: 10.1109/TPAMI.2017.2773081. URL https://arxiv.org/abs/1606.09282. Conference version in ECCV 2016.

Yaron Lipman, Ricky T. Q. Chen, Heli Ben-Hamu, Maximilian Nickel, and Matthew Le. Flow matching for generative modeling. In *The Eleventh International Conference on Learning Representations*, 2023. URL https://openreview.net/forum?id=PqvMRDCJT9t.

Bo Liu, Yifeng Zhu, Chongkai Gao, Yihao Feng, Qiang Liu, Yuke Zhu, and Peter Stone. LIBERO: Benchmarking knowledge transfer for lifelong robot learning. In *Advances in Neural Information Processing Systems 36 (NeurIPS 2023), Datasets and Benchmarks Track*, volume 36, pp. 44776–44791, 2023. URL https://arxiv.org/abs/2306.03310.

Xialei Liu, Chenshen Wu, Mikel Menta, Luis Herranz, Bogdan Raducanu, Andrew D. Bagdanov, Shangling Jui, and Joost van de Weijer. Generative feature replay for class-incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, pp. 226–227, 2020. URL https://openaccess.thecvf.com/content_CVPRW_2020/html/w15/Liu_Generative_Feature_Replay_for_Class-Incremental_Learning_CVPRW_2020_paper.html.

Zuxin Liu, Jesse Zhang, Kavosh Asadi, Yao Liu, Ding Zhao, Shoham Sabach, and Rasool Fakoor. TAIL: Task-specific adapters for imitation learning with large pretrained models. In *The Twelfth International Conference on Learning Representations (ICLR)*, 2024. URL https://arxiv.org/abs/2310.05905.

David Lopez-Paz and Marc'Aurelio Ranzato. Gradient episodic memory for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/f87522788a2be2d171666752f97ddebb-Abstract.html.

James MacQueen. Some methods for classification and analysis of multivariate observations. In *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability*, volume 1, pp. 281–297. University of California Press, 1967.

Arun Mallya and Svetlana Lazebnik. PackNet: Adding multiple tasks to a single network by iterative pruning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 7765–7773, 2018. URL https://openaccess.thecvf.com/content_cvpr_2018/html/Mallya_PackNet_Adding_Multiple_CVPR_2018_paper.html.

Sergi Masip, Pau Rodriguez, Tinne Tuytelaars, and Gido M. van de Ven. Continual learning of diffusion models with generative distillation. In *Proceedings of the 3rd Conference on Lifelong Learning Agents (CoLLAs)*, volume 274 of *Proceedings of Machine Learning Research*, pp. 431–456. PMLR, 2024. URL https://proceedings.mlr.press/v274/masip25a.html.

Maxime Oquab, Timothée Darcet, Théo Moutakanni, Huy V. Vo, Marc Szafraniec, Vasil Khalidov, Pierre Fernandez, Daniel Haziza, Francisco Massa, Alaaeldin El-Nouby, Mido Assran, Nicolas Ballas, Wojciech Galuba, Russell Howes, Po-Yao Huang, Shang-Wen Li, Ishan Misra, Michael Rabbat, Vasu Sharma, Gabriel Synnaeve, Hu Xu, Herve Jegou, Julien Mairal, Patrick Labatut, Armand Joulin, and Piotr Bojanowski. DINOv2: Learning robust visual features without supervision. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://arxiv.org/abs/2304.07193.

German I Parisi, Ronald Kemker, Jose L Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural networks*, 113:54–71, 2019.

William Peebles and Saining Xie. Scalable diffusion models with transformers. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 4195–4205, Paris, France, October 2023. doi: 10.1109/ICCV51070.2023.00387. URL https://arxiv.org/abs/2212.09748.

Lorenzo Pellegrini, Gabriele Graffieti, Vincenzo Lomonaco, and Davide Maltoni. Latent replay for real-time continual learning. In *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, pp. 10203–10209. IEEE, 2020. doi: 10.1109/IROS45743.2020.9341460. URL https://arxiv.org/abs/1912.01100.

Grégoire Petit, Adrian Popescu, Hugo Schindler, David Picard, and Bertrand Delezoide. FeTrIL: Feature translation for exemplar-free class-incremental learning. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pp. 3911–3920, 2023. URL https://openaccess.thecvf.com/content/WACV2023/html/Petit_FeTrIL_Feature_Translation_for_Exemplar-Free_Class-Incremental_Learning_WACV_2023_paper.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In *Proceedings of the 38th International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pp. 8748–8763. PMLR, 2021. URL https://proceedings.mlr.press/v139/radford21a.html.

Sylvestre-Alvise Rebuffi, Alexander Kolesnikov, Georg Sperl, and Christoph H. Lampert. iCaRL: Incremental classifier and representation learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2001–2010, 2017. URL https://openaccess.thecvf.com/content_cvpr_2017/html/Rebuffi_iCaRL_Incremental_Classifier_CVPR_2017_paper.html.

David Rolnick, Arun Ahuja, Jonathan Schwarz, Timothy Lillicrap, and Gregory Wayne. Experience replay for continual learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, 2019. URL https://proceedings.neurips.cc/paper_files/paper/2019/hash/fa7cdfad1a5aaf8370ebeda47a1ff1c3-Abstract.html.

Ralf Römer, Yi Zhang, Yuming Li, and Angela P. Schoellig. CLARE: Continual learning for vision-language-action models via autonomous adapter routing and expansion. *IEEE Robotics and Automation Letters*, 11(7):8331–8338, July 2026. doi: 10.1109/LRA.2026.3693992. URL https://arxiv.org/abs/2601.09512.

Kaushik Roy, Akila Dissanayake, Brendan Tidd, and Peyman Moghadam. M2Distill: Multi-modal distillation for lifelong imitation learning. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 1429–1435, Atlanta, GA, USA, 2025. doi: 10.1109/ICRA55743.2025.11128857. URL https://arxiv.org/abs/2410.00064.

Hanul Shin, Jung Kwon Lee, Jaehong Kim, and Jiwon Kim. Continual learning with deep generative replay. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, 2017. URL https://proceedings.neurips.cc/paper/2017/hash/0efbe98067c6c73dba1250d2beaa81f9-Abstract.html.

James Smith, Yen-Chang Hsu, Jonathan Balloch, Yilin Shen, Hongxia Jin, and Zsolt Kira. Always be dreaming: A new approach for data-free class-incremental learning. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 9374–9384, 2021. URL https://openaccess.thecvf.com/content/ICCV2021/html/Smith_Always_Be_Dreaming_A_New_Approach_for_Data-Free_Class-Incremental_Learning_ICCV_2021_paper.html.

James Seale Smith, Yen-Chang Hsu, Lingyu Zhang, Ting Hua, Zsolt Kira, Yilin Shen, and Hongxia Jin. Continual diffusion: Continual customization of text-to-image diffusion with C-LoRA. *Transactions on Machine Learning Research*, 2024. ISSN 2835-8856. URL https://openreview.net/forum?id=TZdEgwZ6f3.

Weikang Wan, Yifeng Zhu, Rutav Shah, and Yuke Zhu. LOTUS: Continual imitation learning for robot manipulation through unsupervised skill discovery. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 537–544, Yokohama, Japan, 2024. doi: 10.1109/ICRA57147.2024.10611129. URL https://arxiv.org/abs/2311.02058.

Yuzhu Wang, Lechao Cheng, Manni Duan, Yongheng Wang, Zunlei Feng, and Shu Kong. Improving knowledge distillation via regularizing feature direction and norm. In *Computer Vision – ECCV 2024*, volume 15082 of *Lecture Notes in Computer Science*, pp. 20–37. Springer, 2024. doi: 10.1007/978-3-031-72691-0_2. URL https://arxiv.org/abs/2305.17007.

Yue Wu, Yinpeng Chen, Lijuan Wang, Yuancheng Ye, Zicheng Liu, Yandong Guo, and Yun Fu. Large scale incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 374–382, 2019. URL https://openaccess.thecvf.com/content_CVPR_2019/html/Wu_Large_Scale_Incremental_Learning_CVPR_2019_paper.html.

Hongxu Yin, Pavlo Molchanov, Jose M Alvarez, Zhizhong Li, Arun Mallya, Derek Hoiem, Niraj K Jha, and Jan Kautz. Dreaming to distill: Data-free knowledge transfer via deepinversion. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 8712–8721. IEEE, 2020.

Fanqi Yu, Matteo Tiezzi, Tommaso Apicella, Cigdem Beyan, and Vittorio Murino. Lifelong imitation learning with multimodal latent replay and incremental adjustment. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6740–6749, 2026. URL https://arxiv.org/abs/2603.10929.

Michał Zając, Kamil Deja, Anna Kuzina, Jakub M. Tomczak, Tomasz Trzciński, Florian Shkurti, and Piotr Miłoś. Exploring continual learning of diffusion models. arXiv preprint arXiv:2303.15342, 2023. URL https://arxiv.org/abs/2303.15342. Presented at the 4th CLVision Workshop at CVPR 2023 (non-archival track).

Friedemann Zenke, Ben Poole, and Surya Ganguli. Continual learning through synaptic intelligence. In *Proceedings of the 34th International Conference on Machine Learning (ICML)*, volume 70 of *Proceedings of Machine Learning Research*, pp. 3987–3995. PMLR, 2017. URL https://proceedings.mlr.press/v70/zenke17a.html.

Mengyao Zhai, Lei Chen, Frederick Tung, Jiawei He, Megha Nawhal, and Greg Mori. Lifelong GAN: Continual learning for conditional image generation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pp. 2759–2768, 2019. URL https://openaccess.thecvf.com/content_ICCV_2019/html/Zhai_Lifelong_GAN_Continual_Learning_for_Conditional_Image_Generation_ICCV_2019_paper.html.

Fei Zhu, Xu-Yao Zhang, Chuang Wang, Fei Yin, and Cheng-Lin Liu. Prototype augmentation and self-supervision for incremental learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 5871–5880, 2021. URL https://openaccess.thecvf.com/content/CVPR2021/html/Zhu_Prototype_Augmentation_and_Self-Supervision_for_Incremental_Learning_CVPR_2021_paper.html.

APPENDIX

TABLE OF CONTENTS

Additional evaluations, derivations, algorithms, and implementation details.

- (Sec. A) Related Work
- (Sec. B) Additional three-dimensional rollout experiments
- (Sec. C) Shared-velocity floor and condition-specific behavior
  - (Sec. C.1) A shared-velocity reference
- (Sec. D) Final-stage success under the CLARE protocol
- (Sec. E) Magnitude-Aware Regularization: derivation
- (Sec. F) Local state-to-feature maps by ridge regression
- (Sec. G) Sensitivity to the number of TSM clusters
- (Sec. H) Algorithm
  - (Sec. H.1) Efficient Training with Task-Parallel Anchoring
- (Sec. I) Additional evaluation under the CLARE protocol
- (Sec. J) Implementation details
  - (Sec. J.1) Backbone and LIBERO representations
  - (Sec. J.2) Lifelong protocols
  - (Sec. J.3) Optimization and training settings
  - (Sec. J.4) Baseline results and attribution
  - (Sec. J.5) Anchoring and coordinate queries
  - (Sec. J.6) Weights used in training
  - (Sec. J.7) Loss reduction and preservation coefficient
  - (Sec. J.8) Cluster and local-map settings
  - (Sec. J.9) Evaluation counts and diagnostic settings
  - (Sec. J.10) Per-cluster Gaussian coverage
  - (Sec. J.11) State grouping and feature variance
  - (Sec. J.12) MAR diagnostic: evaluation data
  - (Sec. J.13) Transition concentration (panel a)
  - (Sec. J.14) Angular-alignment coefficients (panel b)
  - (Sec. J.15) Hardware evaluation
  - (Sec. J.16) Real robot tasks and evaluation
    - (Sec. J.16.1) Platform, data, and control
    - (Sec. J.16.2) Training and inference
    - (Sec. J.16.3) Task 1: Ring
    - (Sec. J.16.4) Task 2: Box
    - (Sec. J.16.5) Task 3: Carrot
    - (Sec. J.16.6) Task 4: Cup

A Related Work

Continual learning and distillation. Continual learning preserves prior behavior through parameter constraints (Kirkpatrick et al., 2017; Zenke et al., 2017), output distillation (Li & Hoiem, 2018), replay (Rebuffi et al., 2017; Lopez-Paz & Ranzato, 2017; Chaudhry et al., 2019; Rolnick et al., 2019), or parameter allocation (Mallya & Lazebnik, 2018). Functional methods must also choose where to match previous responses: Learning without Forgetting uses new inputs, whereas Dark Experience Replay uses stored examples and logits (Buzzega et al., 2020). Continual generative models likewise use distillation (Zając et al., 2023; Smith et al., 2024), including Lifelong GAN's Montage and Swap inputs (Zhai et al., 2019) and replay images generated by unconditional diffusion teachers (Masip et al., 2024). For conditional action generation, this raises the additional problem of supplying past scene, state, and instruction inputs to the teacher.

Reconstructing past inputs. Reconstructing these inputs connects our setting to generative replay (Shin et al., 2017; Gao et al., 2021), network inversion for data-free distillation (Yin et al., 2020; Smith et al., 2021; Gao et al., 2022), and compressed representations such as latent replay (Pellegrini et al., 2020; Liu et al., 2020) and prototypes (Zhu et al., 2021; Petit et al., 2023). TSM reconstructs paired states and features from aggregate statistics and local maps, without retaining individual features or training an image generator. Together with teacher-derived action coordinates, these conditions support functional matching against latest-task bias in action generation, complementing work on classification recency bias (Wu et al., 2019; Hou et al., 2019).

Continual robot learning. Continual robot learning benchmarks include LIBERO (Liu et al., 2023) and LOTUS (Wan et al., 2024). TAIL uses task-specific adapters (Liu et al., 2024), and CLARE routes and expands adapters (Römer et al., 2026). M2Distill preserves multimodal representations and policy distributions (Roy et al., 2025), MLR-IFA replays latents and actions (Yu et al., 2026), and ReGen generates trajectories with a world–action model (Govind et al., 2026). TETHER instead combines aggregate state–feature statistics with the existing action teacher to construct fresh anchor inputs for one shared policy.

B Additional three-dimensional rollout experiments

Figure 8 compares environment rollouts during sequential learning and under varied task conditions. Panels (a–c) track task 0 after learning LIBERO-Spatial tasks 0–2 in order; panels (d–f) compare responses to interpolated conditions. These environment trajectories complement the action-space denoising paths in Figure 1.

In panels (a–c), task 0 conditions increasingly elicit the newly learned behavior. In panels (d–f), sequential fine-tuning remains near the task 3 reference as conditions vary. ER and TETHER respond to the interpolated conditions, although some ER paths depart from the reference trajectories. Together, these comparisons illustrate latest-task bias in sequential fine-tuning and retained sensitivity to task conditions with TETHER.

C Shared-velocity floor and condition-specific behavior

C.1 A shared-velocity reference

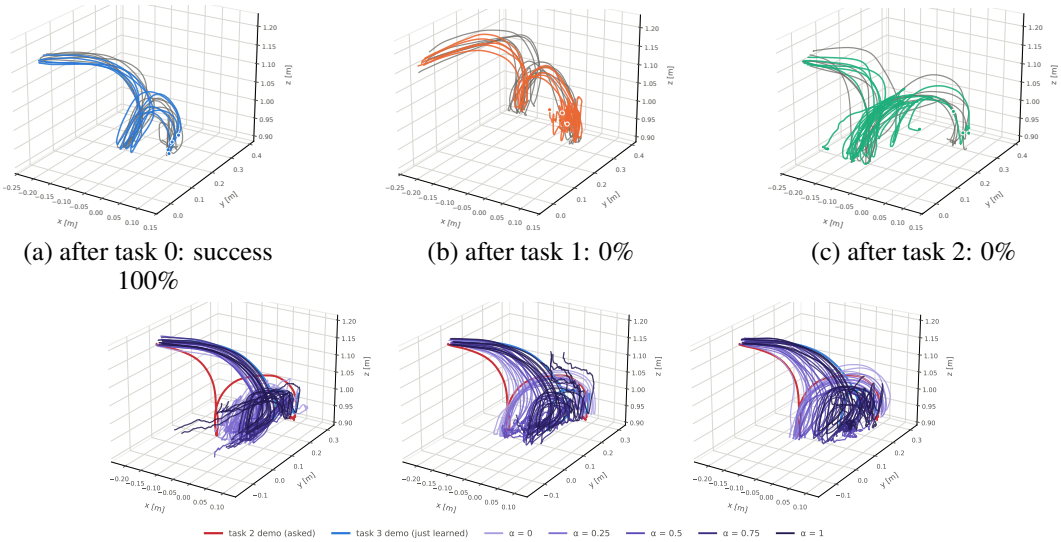
The rollouts in Appendix B motivate a reference for condition blindness: a model that returns the same velocity under different task conditions at a fixed noisy action and flow time. This constraint is

$$v_\theta(x_t, t \mid c_j) = u(x_t, t) \qquad \text{for every evaluated condition } c_j. \tag{15}$$

The common response can depend on (x_t, t), which is held fixed when conditions are compared.

At that coordinate, let $v^\star = a - \varepsilon$ be the current-task flow-matching target from Equation (1), and let $v_{T,j} = v_T(x_t, t \mid c_j)$ be the frozen teacher's target under past task j's condition. With $N = |\mathcal{T}_{\text{past}}|$, define the plain joint squared error of a shared response as

$$L(u) = \mathbb{E}_{h,I}\left[(u_{h,I} - v^\star_{h,I})^2 + \lambda \sum_{j \in \mathcal{T}_{\text{past}}} (u_{h,I} - v_{T,j,h,I})^2\right]. \tag{16}$$



(a) after task 0: success 100%
(b) after task 1: 0%
(c) after task 2: 0%
(d) sequential fine-tuning
(e) experience replay
(f) TETHER

Figure 8: **Three-dimensional rollouts and condition interpolation.** (a–c) After each sequential fine-tuning stage on LIBERO-Spatial tasks 0–2, the policy is evaluated with task 0's condition. Colored curves are policy rollouts; gray curves are demonstrations of the task learned at that stage. Panel labels report task 0 success rates. (d–f) Sequential fine-tuning, ER, and TETHER are evaluated under the displayed interpolation conditions for tasks 2 and 3. The supplied conditions vary in observation and instruction. Red and blue curves are task 2 and task 3 demonstrations, respectively; the other curves are rollouts at the displayed interpolation values.

The current-target coefficient is one, as in Equation (14), and $\lambda > 0$ is the past-task coefficient for this diagnostic. The expectation averages over action steps and components, as in Equation (2). This reference uses plain squared errors.

Setting the derivative with respect to each response component to zero gives

$$u^\star = \frac{v^\star + \lambda \sum_{j \in \mathcal{T}_{\text{past}}} v_{T,j}}{1 + \lambda N}. \tag{17}$$

Completing the square yields

$$L(u) = L(u^\star) + (1 + \lambda N)\mathbb{E}_{h,I}[(u_{h,I} - u^\star_{h,I})^2].$$

The second term is nonnegative and vanishes at $u = u^\star$. Substituting this weighted mean into Equation (16) therefore gives the minimum loss achievable by a shared response:

$$L^\star = L(u^\star) = \mathbb{E}_{h,I}\left[(u^\star_{h,I} - v^\star_{h,I})^2 + \lambda \sum_{j \in \mathcal{T}_{\text{past}}} (u^\star_{h,I} - v_{T,j,h,I})^2\right]. \tag{18}$$

Table 4 reports the LIBERO-Goal comparison under the LOTUS protocol, using all dataset episodes and the TETHER model trained with seed 42. Losses are measured at the end of each reported lifelong stage; the reported preservation coefficient is 3. Other training settings are given in Appendix J. The recorded model losses are below their corresponding shared-response reference values.

A matched comparison uses identical targets, a common (x_t, t) across conditions within each query bundle, the same task weights, and the same plain component-averaged reduction for the model loss and floor. Under this matching, an average loss below the average floor establishes different conditional responses in at least some evaluated bundles. The table provides a numerical reference alongside environment rollouts and final-stage task success in Appendix D.

Table 4: **Single-velocity floor versus measured loss** (LIBERO-Goal, LOTUS protocol). The floor (18) is the smallest loss (16) attainable if all task conditions shared a single velocity at the same coordinate. The measured loss of TETHER is only 4–9% of the floor, indicating condition-dependent velocities.

Stage	Floor $\mathbb{E}[L^\star]$	TETHER loss	Gap	Loss / floor [%]
stage 1	1.20	0.11	1.09	9.2
stage 2	1.07	0.09	0.98	8.4
stage 3	1.36	0.06	1.30	4.4
stage 4	1.50	0.07	1.43	4.7

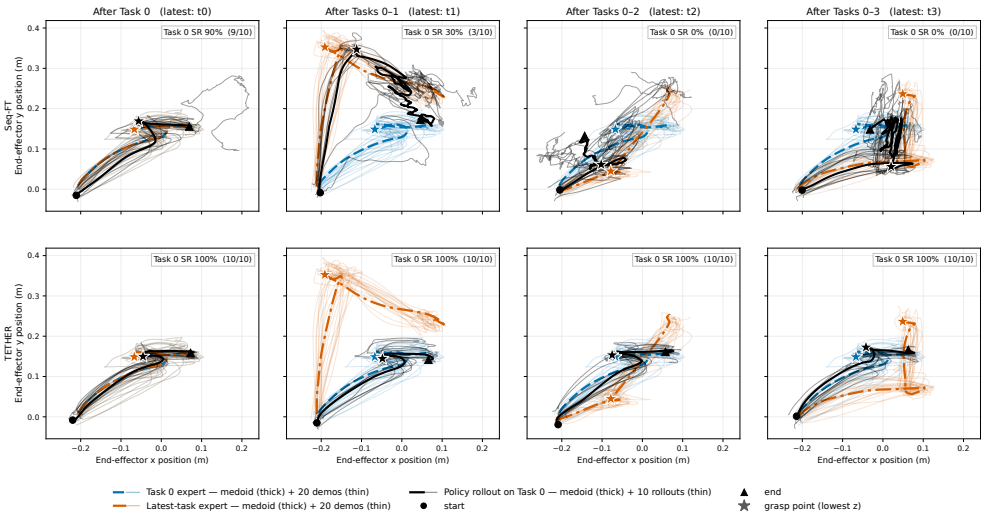


Figure 9: **End-effector trajectories during sequential learning of tasks 0–3.** The horizontal and vertical axes show the absolute x and y coordinates of the end effector. Blue reference curves represent task 0 and orange reference curves represent the latest learned task. In each reference bundle, the thick curve is the expert medoid and the lighter curves are 20 demonstrations.

D FINAL-STAGE SUCCESS UNDER THE CLARE PROTOCOL

Table 5 reports final-stage success after the ten-task LIBERO-Spatial sequence under the CLARE protocol (Römer et al., 2026). ER, CLARE, and TETHER are our runs on the same backbone, initialized from a LIBERO-90 checkpoint pretrained for 200,000 updates. Task entries average success rates over training seeds 7, 42, and 1000. The final column first averages equally over the ten tasks within each seed, then reports the mean and standard deviation across seeds.

TETHER has the highest reported mean on each task and the highest overall final-stage mean: $92.3 \pm 0.8\%$, compared with $79.1 \pm 7.7\%$ for ER and $77.4 \pm 2.5\%$ for CLARE. These results evaluate retained task success after the complete sequence and complement the sequence-wide metrics.

E MAGNITUDE-AWARE REGULARIZATION: DERIVATION

Angular dependence. At a shared anchor query, the pose proxies in Equation (8) are $a_{S,h,\mathrm{pose}} = (v_\theta + \varepsilon)_{h,\mathrm{pose}}$ and $a_{T,h,\mathrm{pose}} = (v_T + \varepsilon)_{h,\mathrm{pose}}$. Their difference is $r_{h,\mathrm{pose}}$. For nonzero pose proxies, let their magnitudes be $\rho_{S,h}$ and $\rho_{T,h}$ and their angle be α_h. Expanding their squared distance gives

$$\|r_{h,\mathrm{pose}}\|_2^2 = \rho_{S,h}^2 + \rho_{T,h}^2 - 2\rho_{S,h}\rho_{T,h}\cos\alpha_h.$$

Multiplying this identity by the final weight w_h in Equation (12) gives Equation (13) directly. The signed cosine coefficient is $-2w_h\rho_{S,h}\rho_{T,h}$. For fixed magnitudes, increasing $\cos\alpha_h$ lowers the loss,

Table 5: **Final-stage success on LIBERO-Spatial under the CLARE protocol.** All methods use the same backbone and LIBERO-90 pretrained initialization. Task entries are means over training seeds 7, 42, and 1000. The last column is the mean ± standard deviation of each seed's average over the ten tasks. All values are percentages; bold marks the largest reported mean in each column.

Method	t_0	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	Avg. ± std
ER	95.0	77.3	85.3	89.3	69.7	52.7	72.0	74.7	90.0	84.7	79.1±7.7
CLARE	92.3	82.7	95.0	92.0	66.0	48.0	81.3	65.7	79.7	71.7	77.4±2.5
TETHER (ours)	**99.3**	**96.3**	**99.0**	**97.3**	**83.3**	**90.7**	**94.7**	**76.0**	**94.0**	**92.0**	**92.3±0.8**

so the loss favors aligned directions. Holding the magnitudes and teacher-derived weight fixed, its angular derivative is

$$\frac{\partial\big(w_h\|r_{h,\mathrm{pose}}\|_2^2\big)}{\partial\alpha_h} = 2w_h\rho_{S,h}\rho_{T,h}\sin\alpha_h.$$

The positive factor $w_h\rho_{S,h}\rho_{T,h}$ in Figure 4(b) is half the magnitude of the cosine coefficient. It describes the loss emphasis on angular agreement, with the common factor of two omitted from both curves.

Magnitude and directional errors. Adding and subtracting $2\rho_{S,h}\rho_{T,h}$ in the unweighted expansion and then multiplying by w_h yields

$$w_h\|r_{h,\mathrm{pose}}\|_2^2 = w_h(\rho_{S,h}-\rho_{T,h})^2 + 2w_h\rho_{S,h}\rho_{T,h}(1-\cos\alpha_h).$$

The first term measures magnitude mismatch. The second measures angular mismatch and vanishes for aligned directions. MAR applies the same weight to both terms, preserving a single squared-error anchor while emphasizing small teacher responses.

Relative-magnitude interpretation. For $\rho_{T,h} > 0$, dividing the unweighted residual by the squared teacher magnitude gives

$$\frac{\|r_{h,\mathrm{pose}}\|_2^2}{\rho_{T,h}^2} = \left(\frac{\rho_{S,h}}{\rho_{T,h}}-1\right)^2 + 2\frac{\rho_{S,h}}{\rho_{T,h}}(1-\cos\alpha_h).$$

The first term measures relative magnitude error. The second measures directional error, scaled by the student-to-teacher magnitude ratio. For equal magnitudes, the latter becomes $2(1-\cos\alpha_h)$. Thus, the same absolute magnitude deviation represents a larger relative change for a smaller teacher response, motivating an emphasis on magnitude preservation as well as direction.

The implemented loss weights this relative-error decomposition by $w_h\rho_{T,h}^2$:

$$w_h\|r_{h,\mathrm{pose}}\|_2^2 = w_h\rho_{T,h}^2\left[\left(\frac{\rho_{S,h}}{\rho_{T,h}}-1\right)^2 + 2\frac{\rho_{S,h}}{\rho_{T,h}}(1-\cos\alpha_h)\right].$$

This is an identity for the existing anchor, not an additional relative-error or angular loss.

Reference scale and batch normalization. The task reference $\bar{\rho}_j^2$ is the median squared pose-action magnitude defined in Section 3.2. The positive reference bounds the unnormalized inverse weight $\widetilde{w}_h = 1/(\rho_{T,h}^2+\bar{\rho}_j^2)$ and moderates weight contrasts. Dividing by its mean over all examples and action steps of the same past-task batch gives the final w_h, whose batch mean is one. Equation (23) writes this normalization with the example index b made explicit, whereas Equation (12) suppresses that index. This normalization preserves relative weights but does not fix the mean weighted residual loss.

For a fixed finite batch and finite student magnitude, the stabilized weight remains finite as $\rho_{T,h}$ approaches zero. The angular coefficient $2w_h\rho_{S,h}\rho_{T,h}$ still approaches zero, consistent with the absence of a defined teacher direction at zero magnitude. The rationale concerns reweighting nonzero small responses, including the range shown in Figure 4(b).

F LOCAL STATE-TO-FEATURE MAPS BY RIDGE REGRESSION

Ridge regression fits a linear map while penalizing its squared size. With input samples as columns of X and corresponding output samples as columns of Y, its multivariate form is

$$\widehat{B} = \arg\min_{B} \{ \|Y - BX\|_F^2 + \gamma \|B\|_F^2 \},$$

where $\| \cdot \|_F$ is the Frobenius norm and γ is the ridge coefficient. We apply this fit within each state cluster of each task, using the notation of Equation (4).

For task j, standardize states using the task statistics $z_{j,d}(s) = (s_d - \bar{s}_{j,d})/\bar{\sigma}_{j,d}$ and cluster these standardized states. Let $\mathcal{I}_{j,k}$ index the frames assigned to cluster k, and let $n_{j,k} = |\mathcal{I}_{j,k}|$. The cluster stores its frame frequency $\pi_{j,k} = n_{j,k} / \sum_r n_{j,r}$, raw-state mean $\mu^s_{j,k}$ and covariance $\Sigma_{j,k}$, and mean visual feature $\mu^o_{j,k}$. Its standardized state mean is $\bar{z}_{j,k} = z_j(\mu^s_{j,k})$.

In the ridge objective, set

$$X_{j,k} = [\, z_j(s_i) - \bar{z}_{j,k} \,]_{i \in \mathcal{I}_{j,k}},$$
$$Y_{j,k} = [\, o_i - \mu^o_{j,k} \,]_{i \in \mathcal{I}_{j,k}}, \qquad B = A_{j,k}.$$

Thus $X_{j,k} \in \mathbb{R}^{d_s \times n_{j,k}}$, $Y_{j,k} \in \mathbb{R}^{d_o \times n_{j,k}}$, and $A_{j,k} \in \mathbb{R}^{d_o \times d_s}$ is the map stored in $\mathcal{M}_j$. Define

$$C_{oz,j,k} = Y_{j,k} X_{j,k}^\top,$$
$$C_{zz,j,k} = X_{j,k} X_{j,k}^\top, \qquad \gamma_{j,k} = \eta \ \mathrm{tr}(C_{zz,j,k})/d_s.$$

Here η is the dimensionless ridge setting and tr denotes the matrix trace.

The cluster-specific ridge objective is

$$F(A) = \sum_{i \in \mathcal{I}_{j,k}} \|o_i - \mu^o_{j,k} - A(z_j(s_i) - \bar{z}_{j,k})\|_2^2 + \gamma_{j,k} \|A\|_F^2.$$

Its derivative is

$$\nabla_A F = 2(A C_{zz,j,k} - C_{oz,j,k}) + 2\gamma_{j,k} A.$$

Setting it to zero gives $A(C_{zz,j,k} + \gamma_{j,k} \mathbf{I}_{d_s}) = C_{oz,j,k}$, and hence

$$A_{j,k} = C_{oz,j,k} (C_{zz,j,k} + \gamma_{j,k} \mathbf{I}_{d_s})^{-1}. \tag{19}$$

For positive $\gamma_{j,k}$, the matrix is positive definite and the solution is unique. The resulting predictor is $\mu^o_{j,k} + A_{j,k}(z_j(s) - \bar{z}_{j,k})$, the feature map used in Equation (4). Small clusters use $A_{j,k} = 0$. Cluster thresholds, state-scale stabilization, and the ridge setting are specified in Appendix J.

For completeness, the cluster prior, Gaussian state model, and local feature predictor define the joint approximation

$$\begin{aligned} \hat{p}_j(s, o, k) &= \hat{p}_j(k) \hat{p}_j(s \mid k) \hat{p}_j(o \mid s, k) \\ &= \pi_{j,k} \, \mathcal{N}(s; \mu^s_{j,k}, \Sigma_{j,k}) \, \delta_{\mu^o_{j,k} + A_{j,k}(z_j(s) - \bar{z}_{j,k})}(o), \\ \hat{p}_j(s, o) &= \sum_k \hat{p}_j(s, o, k). \end{aligned}$$

Here δ is a point mass at the predicted feature. Equation (4) samples this joint model by drawing a cluster, a state from that cluster, and its paired feature from the local map.

G SENSITIVITY TO THE NUMBER OF TSM CLUSTERS

Experimental setup. We vary only the number of clusters in Task Statistics Memory (TSM), $K \in \{20, 40, 70, 100, 120, 140\}$, while keeping all other settings fixed. The policy learns ten LIBERO-Goal tasks sequentially, using 50 demonstrations and 10,000 training steps per task. Each task–stage success-rate entry is evaluated with 100 rollouts using the fixed evaluation seed 7. All six configurations use training seed 42. We report FWT, NBT, and AUC using the same metric definitions as the other continual-learning evaluations. Higher FWT and AUC and lower NBT are better.

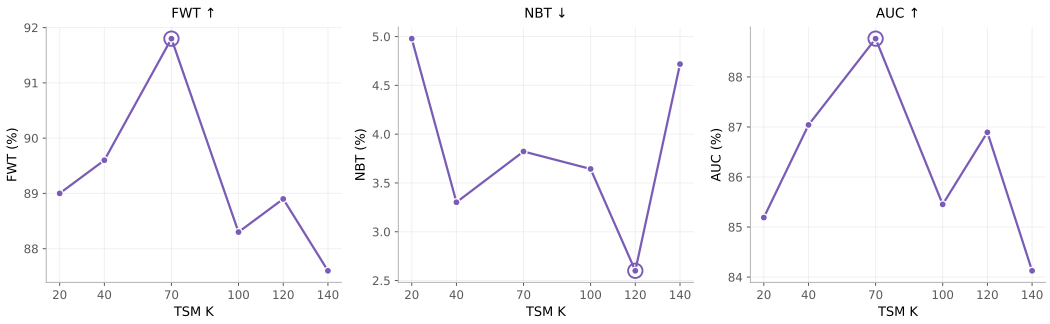


Figure 10: **Sensitivity to the number of TSM clusters K on LIBERO-Goal.** Ten sequential tasks, 10,000 training steps per task, and 100 rollouts per task–stage entry, using a single training seed (42). Open circles mark the best tested K for each metric: $K = 70$ for FWT/AUC and $K = 120$ for NBT. All three metrics deteriorate at $K = 140$ relative to $K = 120$.

Results and interpretation. Figure 10 shows that the preferred number of TSM clusters depends on the metric. Among the six tested values, $K = 70$ gives the highest FWT and AUC, whereas $K = 120$ gives the lowest NBT. FWT and AUC increase from $K = 20$ through $K = 40$ to $K = 70$, drop at $K = 100$, and partially recover at $K = 120$ before reaching their lowest values at $K = 140$. NBT also varies nonmonotonically, with the highest value at $K = 20$ and the lowest at $K = 120$. Thus, finer partitions do not consistently improve continual-learning performance, and the setting with the least forgetting differs from the setting with the highest acquisition and overall sequence performance in this run.

With a fixed demonstration budget, increasing K reduces the average number of frames per cluster. Less reliable estimates of the cluster means and covariances $(\mu^s_{j,k}, \Sigma_{j,k})$ are one possible explanation for the deterioration at $K = 140$ relative to $K = 120$, but this sweep does not establish the cause. These results characterize a single training seed rather than an optimum validated across seeds.

H ALGORITHM

Our code will be released as open source upon acceptance of the paper

Algorithm 1 One learning stage of TETHER. With no past tasks, the base stage uses current-task flow matching and then constructs TSM for the new tasks. When a stage introduces several tasks (LOTUS protocol, Appendix J.2), $\mathcal{T}_{\text{new}}$ denotes the set of their indices and $\mathcal{D}_{\text{new}}$ their pooled demonstrations; Section 2.1 writes the single-task case $\mathcal{T}_{\text{new}} = n$.

Require: previous policy parameters $\theta^{(\text{prev})}$; new demonstrations $\mathcal{D}_{\text{new}}$; past-task instructions and TSM $\{(\ell_j, \mathcal{M}_j)\}_{j \in \mathcal{T}_{\text{past}}}$

load the past-task memories and their corresponding instructions
initialize $\theta \leftarrow \theta^{(\text{prev})}$ and freeze $v_T \leftarrow v_{\theta^{(\text{prev})}}$
for each training step **do**
  clear student gradients
  sample a batch from $\mathcal{D}_{\text{new}}$; compute and back-propagate $\mathcal{L}_{\text{FM}}$ by (2)
  for each past task $j \in \mathcal{T}_{\text{past}}$ **do**
    sample paired $(\tilde{s}_j, \tilde{o}_j)$ using $\mathcal{M}_j$ by (4)
    construct $\hat{a}_j$ and $\hat{x}_t$ by (5) under ℓ_j; retain the anchor noise ε
    evaluate teacher and student at the shared query; compute r_j by (6) and the teacher pose magnitudes $\rho_{T,h}$ by (9)
    compute the detached weights w_h from $\rho_{T,h}$ and $\bar{\rho}_j^2$ by (23), the batch-wise form of (12)
    compute the past-task term of (14), $\mathcal{L}_j = \mathbb{E}_{q \sim \mathcal{Q}_j, h}\big[w_h \|r_{h,\text{pose}}\|_2^2 + \|r_{h,\text{grip}}\|_2^2\big]$, and back-propagate $\lambda \mathcal{L}_j$
  end for
  update the student parameters using the accumulated gradients
end for
construct and store $\mathcal{M}_j$ for each $j \in \mathcal{T}_{\text{new}}$, including the local maps from (19) and the reference pose magnitude $\bar{\rho}_j^2$; retain the instruction ℓ_j
retain the current policy as the next stage's teacher; discard the current stage's individual demonstrations

H.1 EFFICIENT TRAINING WITH TASK-PARALLEL ANCHORING

Task-Parallel Anchoring (TPA) in Figure 7 groups velocity-network queries across tasks while preserving the task-wise implementation's queries, weights, and loss reductions. Appendix J specifies the component-mean reduction and its coefficient convention relative to Equation (14). The teacher remains frozen throughout a learning stage. For each past task, TSM supplies paired synthetic states and visual features, which are combined with that task's retained instruction. Current images pass through the frozen visual encoder; past-task queries enter at the feature level. The resulting current and past conditions use the same student velocity network.

Each update requires two sequential teacher evaluations over all past tasks. The first evaluates all past-task inputs at $(x_0, 0)$ to form the one-step estimates $\hat{a}_j$. These estimates determine $\hat{x}_t$ through independent anchor noise ε. Only then can the second evaluation compute all teacher matching targets at $(\hat{x}_t, t)$. Thus TPA combines queries *across tasks*; it does not combine these dependent teacher evaluations into one forward pass. Both teacher passes disable gradient recording. The student first evaluates and back-propagates the current flow-matching loss. It then evaluates all past-task anchor queries in one concatenated batch with gradients enabled.

Task boundaries are retained when reducing the outputs. Within each past task, we compute the inverse pose weights and their mean normalization over that task's examples and action steps exactly as in Equation (23); weights are detached and the gripper weight remains one. Each $\widehat{\mathcal{L}}_j$ is a component-averaged *per-task* loss. We sum these losses with coefficient λ_{impl} and back-propagate their sum into the accumulated current-task gradients, then perform one optimizer update. Any gradient clipping is applied at the same point as in the task-wise implementation. A global mean over all past-task examples would change the preservation strength, and global weight normalization would change the individual task objectives.

Algorithm 2 One TETHER update with past tasks using TPA. Concatenation and splitting preserve task boundaries; the base stage uses only flow matching.

Require: student v_θ, stage-frozen teacher v_T, current demonstrations, past memories $\{(\mathcal{M}_j, \ell_j)\}_j$

clear student gradients; sample current demonstrations and construct (x_t, t, c) and flow targets by (1), using frozen image features
evaluate the student on (x_t, t, c); compute and back-propagate the current-task loss of (2), averaged over examples, action steps, and components
for each past task j **do**
    sample paired $(\tilde{s}_j, \tilde{o}_j)$ by (4); set $c_j = (\tilde{s}_j, \tilde{o}_j, \ell_j)$
    draw independent standard Gaussian arrays $x_{0,j}, \varepsilon_j$; reuse the current batch times t
end for
concatenate all $(x_{0,j}, 0, c_j)$; evaluate v_T once without gradients and split the outputs u_j
form $\hat{a}_j = x_{0,j} + u_j$ and $\hat{x}_{t,j} = (1-t)\varepsilon_j + t\hat{a}_j$ for every j
concatenate all $(\hat{x}_{t,j}, t, c_j)$; evaluate v_T once without gradients and split the targets y_j
concatenate all past $(\hat{x}_{t,j}, t, c_j)$; evaluate v_θ once with gradients and split outputs
for each past task j **do**
    compute the component-averaged per-task loss $\widehat{\mathcal{L}}_j = \mathcal{L}_j / A$ (Appendix J.7) over that task's examples, action steps, and components
    compute the component-averaged $\widehat{\mathcal{L}}_j$ specified in Appendix J over that task's examples, action steps, and components
end for
back-propagate $\lambda_{\text{impl}} \sum_j \widehat{\mathcal{L}}_j$ into the accumulated gradients; update the student once

For fixed queries and matched random realizations, including dropout, these reductions are algebraically equivalent to the task-wise objective and gradient, up to floating-point differences. Ordinary runs can consume different dropout streams under the two execution orders, so individual updates need not be identical. TPA reduces task-wise execution overhead while increasing simultaneous activation storage. It does not change retained storage: both implementations keep the same TSM and one 175.5 MB teacher snapshot, replaced between stages.

I ADDITIONAL EVALUATION UNDER THE CLARE PROTOCOL

Table 6 reports continual-learning results on LIBERO-Spatial, LIBERO-Long, and LIBERO-Goal under the CLARE protocol (Römer et al., 2026). This setting introduces one task per stage. Our runs use 50 demonstrations per task and 10,000 updates per stage; Appendix J gives the remaining settings.

Table 6: **Lifelong results under the CLARE protocol** on LIBERO-Spatial, LIBERO-Long and LIBERO-Goal. All entries are mean$^{\pm\text{std}}$ over three seeds, including our runs. AUC and FWT: higher is better; NBT: lower is better. SeqFFT, SeqLoRA, PackNet and ER are as reported by Römer et al. (2026); CLARE is re-run by us under the same protocol. Best in bold, second best underlined.

	LIBERO-Spatial			LIBERO-Long			LIBERO-Goal		
Method	AUC↑	NBT↓	FWT↑	AUC↑	NBT↓	FWT↑	AUC↑	NBT↓	FWT↑
SeqFFT	$27.7^{\pm 0.6}$	$94.6^{\pm 0.9}$	$\mathbf{94.7}^{\pm 0.3}$	$22.4^{\pm 0.3}$	$74.7^{\pm 1.1}$	$\underline{76.1}^{\pm 1.0}$	$26.7^{\pm 0.9}$	$95.3^{\pm 1.4}$	$\underline{94.1}^{\pm 0.2}$
SeqLoRA	$27.3^{\pm 1.3}$	$89.2^{\pm 1.3}$	$90.1^{\pm 2.1}$	$21.4^{\pm 1.0}$	$71.6^{\pm 1.6}$	$73.1^{\pm 1.8}$	$26.1^{\pm 0.3}$	$90.8^{\pm 1.7}$	$90.1^{\pm 1.6}$
PackNet	$8.6^{\pm 0.1}$	$60.3^{\pm 0.7}$	$54.7^{\pm 0.5}$	$4.8^{\pm 0.2}$	$41.3^{\pm 1.2}$	$37.2^{\pm 1.0}$	$10.5^{\pm 0.3}$	$67.0^{\pm 1.1}$	$60.3^{\pm 1.0}$
ER	$77.6^{\pm 0.8}$	$20.9^{\pm 2.0}$	$\underline{92.7}^{\pm 1.0}$	$60.5^{\pm 0.2}$	$22.7^{\pm 1.8}$	$\mathbf{76.6}^{\pm 0.9}$	$76.0^{\pm 0.9}$	$25.1^{\pm 0.5}$	$\mathbf{94.4}^{\pm 0.7}$
CLARE	$\underline{79.6}^{\pm 4.4}$	$\underline{-0.3}^{\pm 1.4}$	$78.0^{\pm 4.0}$	$\underline{64.7}^{\pm 2.3}$	$\underline{1.4}^{\pm 1.0}$	$64.3^{\pm 0.5}$	$\mathbf{88.5}^{\pm 1.3}$	$\mathbf{0.3}^{\pm 1.3}$	$87.8^{\pm 0.8}$
TETHER	$\mathbf{90.4}^{\pm 0.9}$	$\mathbf{-1.5}^{\pm 1.0}$	$91.3^{\pm 0.9}$	$\mathbf{69.8}^{\pm 2.7}$	$\mathbf{-1.2}^{\pm 0.8}$	$68.8^{\pm 2.8}$	$\underline{86.8}^{\pm 1.1}$	$\underline{3.9}^{\pm 0.7}$	$89.7^{\pm 0.7}$

CLARE uses autonomous routing and expansion of task-specific adapters (Römer et al., 2026). Against this baseline, TETHER retains one shared policy through velocity regularization and has higher mean AUC and lower mean NBT on Spatial and Long. Its NBT values are -1.5 and -1.2, compared with CLARE's -0.3 and 1.4, respectively. On Goal, CLARE has higher mean AUC and lower mean NBT, while TETHER has higher mean FWT than CLARE. These results support strong retention on Spatial and Long under this additional sequential setting.

J IMPLEMENTATION DETAILS

J.1 BACKBONE AND LIBERO REPRESENTATIONS

We use CLARE's flow-matching base architecture (Römer et al., 2026). For LIBERO, two proprioceptive frames form a state vector with $d_s = 16$. DINOv2 class-token features from two views over those frames give $d_o = 3072$ visual dimensions (Oquab et al., 2024); CLIP embeds the instruction (Radford et al., 2021). Both pretrained encoders remain frozen. Learned linear projections map the modalities into common-dimensional tokens for the transformer velocity network (Peebles & Xie, 2023). The projections and velocity network constitute the trainable parameters θ. Each LIBERO chunk has $H = 16$ steps and $A = 7$ components: six pose deltas and one gripper command. Actions are normalized to $[-1, 1]$; the gripper uses unit-magnitude values. Pose-magnitude calculations exclude the gripper dimension. The real robot uses the six-component absolute-joint representation described below.

J.2 LIFELONG PROTOCOLS

LOTUS protocol. We follow the LOTUS protocol (Wan et al., 2024) on LIBERO-Object, LIBERO-Goal, and LIBERO-50 (Liu et al., 2023). Training consists of a jointly trained base stage followed by lifelong adaptation stages. For LIBERO-Object and LIBERO-Goal, the base stage uses the first six tasks with 50 demonstrations per task. The remaining four tasks are introduced individually over four adaptation stages, each with ten demonstrations for the new task. For LIBERO-50, the base stage uses the first 25 tasks with 50 demonstrations per task. The remaining 25 tasks are introduced over five adaptation stages, with five new tasks per stage and ten demonstrations per

new task. TETHER does not revisit past demonstrations during adaptation. Table 8 summarizes the training settings.

CLARE protocol. Additional evaluations follow the CLARE protocol (Römer et al., 2026) on LIBERO-Spatial, LIBERO-Long, LIBERO-Goal, and LIBERO-40. Tasks are introduced one at a time, using 50 demonstrations per task with no demonstration holdout in the standard benchmark runs. Diagnostic-specific splits and the LIBERO-40 task sequence are specified in Appendix J.9.

Evaluation metrics. We report forward transfer (FWT), negative backward transfer (NBT), and success-rate AUC. To express their definitions, we use the one-task-at-a-time indexing of Section 2.1. Let M denote the total number of tasks, and let $R_{n,j}$ denote the success rate on task $\mathfrak{T}_j$ after learning the first n tasks. Uppercase R distinguishes success rates from the velocity residual r_j. In the grouped LOTUS stages described above, tasks introduced together share a stage-end evaluation checkpoint.

FWT measures success immediately after each task is learned:

$$\mathrm{FWT} = \frac{1}{M}\sum_{j=1}^{M} R_{j,j}. \tag{20}$$

Higher FWT indicates stronger acquisition of new tasks under the specified training budget.

For each task $j < M$, NBT_j measures the average decrease from its initial post-training success rate over subsequent checkpoints. NBT averages these decreases across tasks:

$$\mathrm{NBT}_j = \frac{1}{M-j}\sum_{n=j+1}^{M}\big(R_{j,j} - R_{n,j}\big), \qquad \mathrm{NBT} = \frac{1}{M-1}\sum_{j=1}^{M-1}\mathrm{NBT}_j. \tag{21}$$

Lower NBT indicates less forgetting. Negative NBT indicates that later success rates exceed the initial post-training rates on average.

AUC first averages each task's success rate over checkpoints from its acquisition onward, then averages across tasks:

$$\mathrm{AUC}_j = \frac{1}{M-j+1}\sum_{n=j}^{M} R_{n,j}, \qquad \mathrm{AUC} = \frac{1}{M}\sum_{j=1}^{M}\mathrm{AUC}_j. \tag{22}$$

Higher AUC indicates stronger average success throughout the learning sequence. Evaluation seeds and rollout counts are given in Appendix J.9.

J.3 Optimization and training settings

Tables 7 and 8 summarize the simulation settings, which are shared across suites and methods in our runs. The optimizer and scheduler are reinitialized at each learning-stage boundary; policy parameters carry forward. Training seeds control parameter initialization, data order, and clustering. Evaluation seed 7 fixes rollout initial states across methods.

J.4 Baseline results and attribution

Sequential, ER, and CLARE in Table 1 are our runs on the shared backbone under the LOTUS protocol. BUDS and LOTUS values are reported by Wan et al. (2024); IsCiL, M2Distill, TAIL, MLR, and MLR+IFA values are taken from the comparison compiled by Yu et al. (2026). These reported rows retain their source architectures and training configurations. Under the alternate CLARE protocol, SeqFFT, SeqLoRA, PackNet, and ER in Table 6 are reported by Römer et al. (2026), whereas CLARE and TETHER are our runs. All three methods in Table 5 are our runs on the common backbone. ER in Figures 6, 5, and 7 is also evaluated in our implementation.

J.5 Anchoring and coordinate queries

Each query uses one teacher evaluation to obtain $\hat{a}_j$ and a subsequent teacher evaluation to obtain the matching target at $\hat{x}_t$. The initial action x_0 and anchor noise ε are independent standard Gaussian

Table 7: **Simulation optimization settings.** These settings apply to our runs under both lifelong protocols.

Setting	Value
Optimizer	Adam
Learning rate	10^{-4}
Betas	$(0.95, 0.999)$
Epsilon	10^{-8}
Weight decay	10^{-6}
Schedule	Cosine, with 500 warmup updates
Gradient clipping	Policy preset; nonfinite errors disabled
Automatic mixed precision	Disabled

Table 8: **Simulation training protocols.** LOTUS adaptation stages and update totals exclude the jointly trained base stage. Each listed stage uses 10,000 updates. The CLARE column describes the ten-task suite runs; the LIBERO-40 sequence is specified below.

Setting	CLARE	LOTUS Goal/Object	LOTUS-50
Batch size	32	32	32
Stages	10	4 after base	5 after base
Updates	100,000	40,000 after base	50,000 after base
Demonstrations per task	50	Base 50; new 10	Base 50; new 10
Training seeds	7, 42, 1000	7, 42, 1000	7, 42, 1000
Evaluation seed	7	7	7

action arrays. The initial draw is used only to construct $\hat{a}_j$; the anchor noise is retained for magnitude measurement. The time t is taken from the current flow-matching batch and has marginal distribution $\mathcal{U}(0, 1)$. The reported sampler is `hard_linear`, with `speedinv` anchor weighting and reference multiplier $\kappa = 1$.

J.6 Weights used in training

For a query batch from past task j, let $b \in \{1, \ldots, N_{\text{mini}}\}$ index its examples and $h \in \{1, \ldots, H\}$ its action steps, and let $\rho_{T,b,h}$ be the teacher pose magnitude of Equation (9) for example b. The complete weight calculation is

$$\widetilde{w}_{b,h} = \frac{1}{\rho_{T,b,h}^2 + \bar{\rho}_j^2}, \qquad \bar{w} = \frac{1}{N_{\text{mini}} H} \sum_{b=1}^{N_{\text{mini}}} \sum_{h=1}^{H} \widetilde{w}_{b,h}, \qquad w_{b,h} = \frac{\widetilde{w}_{b,h}}{\bar{w}}. \tag{23}$$

Equation (12) displays the same construction with the example index suppressed: the mean $\bar{w}$ is taken jointly over all examples and action steps of that task's batch, so that the final weights have unit batch mean, $\mathbb{E}_{b,h}[w_{b,h}] = 1$. The weights are detached from student gradients and the gripper weight remains one. Appendix E derives the angular coefficient and the relative-magnitude and directional decomposition of the weighted loss.

J.7 Loss reduction and preservation coefficient

The implementation averages the weighted squared residual over examples, action steps, and all A action components, using $\lambda_{\text{impl}} = 3$ for each past task. The corresponding coefficient for the norm-summed anchor in Equation (14) is $\lambda = \lambda_{\text{impl}}/A$. Past-task losses are summed. The floor diagnostic and its preservation coefficient are described in Appendix C.

J.8 Cluster and local-map settings

For the benchmark runs, we use an initial TSM cluster count of $K = 25$ under the LOTUS protocol and $K = 100$ under the CLARE protocol. These settings account for the different demonstration

budgets for newly introduced tasks: ten episodes per task under LOTUS and 50 under CLARE (Table 8). Frames from all available demonstration episodes of a task are pooled, and a cluster can contain frames from several episodes. After merging small clusters as described below, K_j denotes the number of retained clusters for task j. Diagnostic-specific cluster counts are reported in their respective settings below. The dimensionless ridge coefficient is $\eta = 0.05$, giving $\gamma_{j,k} = \eta \operatorname{tr}(C_{zz,j,k})/d_s$ in Appendix F. Clusters with fewer than 24 frames use $A_{j,k} = 0$. Clusters with fewer than five frames are merged into the nearest cluster in standardized state space. Per-cluster state standard deviations are floored at 10^{-3} of the corresponding task-level standard deviation. Codebook construction uses diagonal state covariances with divisor $n_{j,k}$. Local maps $A_{j,k}$ in TSM $\mathcal{M}_j$ are stored in half precision.

J.9 Evaluation counts and diagnostic settings

The main lifelong comparison reports FWT, NBT, and AUC in the single-checkpoint form used by LOTUS, using three training seeds and 20 rollouts per task and checkpoint. The four-task diagnostic in Figure 1 uses a separate four-task sequence. It visualizes pose-action denoising paths under held-out conditions from tasks 0–3 after sequential learning, using a shared initial noise and 100 Euler steps, followed by projection onto two principal components. Its success annotations refer to the requested task. Both displayed panels remove the across-condition mean: sequential fine-tuning is on the left and TETHER on the right. Centering removes a common component; by itself it does not change pairwise differences between points from which the same mean is subtracted. The small spread of sequential fine-tuning's condition-specific means indicates weak differentiation in the displayed projection, rather than exact equality of the full velocity fields. The plotted paths are generated by integrating the velocity network; sharing the initial noise does not force different conditions to follow the same noisy-action coordinates at later steps.

Figure 5 compares ER and TETHER on LIBERO-40 under the CLARE protocol. The sequence learns LIBERO-10, LIBERO-Goal, LIBERO-Spatial, and LIBERO-Object in that order, starting from a LIBERO-90 pretrained checkpoint. Table 2 reports three-seed results for anchor-input variants. Shuffled draws four observation–state pairs and permutes their associations; Coupled retains those associations. Full TETHER also uses MAR with the weight construction above.

J.10 Per-cluster Gaussian coverage

Figure 3(a) evaluates diagonal Gaussian state models on 3,677 clusters containing 317,386 frames from LIBERO-Spatial, LIBERO-Goal, LIBERO-Object, and LIBERO-10. The respective cluster counts are 888, 861, 937, and 991. Each suite contains ten tasks; the initial clustering uses $K = 100$ for 50 demonstrations and state dimension $d_s = 16$. Only clusters with at least 21 frames enter this diagnostic. All frames in each retained cluster are used both to fit and to evaluate its model.

For a cluster with n_k frames, the diagnostic estimates its mean and diagonal sample covariance using divisor $n_k - 1$, with variances floored at 10^{-12}. No covariance ridge term is applied to this diagonal estimate. This diagnostic estimator uses $n_k - 1$, whereas codebook construction uses n_k. Write the diagnostic covariance as $\widehat{\Sigma}_{k,\mathrm{diag}}$ and define

$$D_k^2(s) = (s - \widehat{\mu}_k)^\top \widehat{\Sigma}_{k,\mathrm{diag}}^{-1}(s - \widehat{\mu}_k),$$
$$\widehat{C}_k(p) = \frac{1}{n_k} \sum_{s \in \mathcal{S}_k} \mathbf{1}\{D_k^2(s) \le \tau_{n_k}(p)\}.$$

For each cluster, we draw one standard Gaussian dataset with the same frame count and state dimension, refit its mean and coordinate-wise sample standard deviations, and evaluate the squared radii of those same points. We pool simulated radii across clusters with the same frame count n; $\tau_n(p)$ is their empirical p-quantile. This yields one bootstrap dataset per cluster and 317,386 simulated points overall. The pools contain a minimum of 192, median of 982, and maximum of 3,168 radii. One random-number stream, initialized with the diagnostic seed plus 12345, is used across clusters.

The horizontal axis is expected Gaussian coverage and the vertical axis is observed coverage, both in percent. Dots represent clusters and large markers represent suite medians. The medians show greater coverage near the center and lower coverage in the larger regions than the Gaussian reference. Observed and simulated statistics are both evaluated in sample. The plot describes radial

coverage under the diagonal state model; it does not evaluate held-out coverage or reconstructed-feature validity.

J.11 State grouping and feature variance

Figure 3(b) uses LIBERO-Spatial task 0 with all 50 demonstration episodes. Removing the final seven frames of each episode leaves 4,718 frames; no further subsampling or train–test split is used. For each $K \in \{10, 20, 50, 70, 100, 120, 140\}$, temporal bins use relative episode position: frame index i in an episode of retained length L is assigned to $\min\{\lfloor Ki/L \rfloor, K-1\}$, with i starting at zero. This aligns relative progress without resampling episodes or performing event alignment. State groups use coordinate-wise standardized states and k-means with seed 42 and ten initializations. Groups with fewer than five frames are merged into the nearest group; all displayed partitions retain K groups.

Each visual feature $o_i \in \mathbb{R}^{3072}$ concatenates frozen DINOv2-base class-token features from two cameras over two observation steps, ordered by observation step and camera. Images receive mean–standard-deviation preprocessing; the concatenated features receive no additional scaling, whitening, or normalization. The state used for grouping concatenates two eight-dimensional proprioceptive frames, each min–max normalized to $[-1, 1]$, before coordinate-wise standardization for clustering. The separate joint-state field is unused. For a partition $g(i)$ with group feature mean $m_{g(i)}$, the plotted residual percentage is

$$100 \, \frac{\sum_{d=1}^{d_o} \mathrm{Var}_i[o_{i,d} - m_{g(i),d}]}{\sum_{d=1}^{d_o} \mathrm{Var}_i[o_{i,d}]},$$

where both sample variances use all retained frames and divisor $n-1$. The global reference is 100%. At $K = 100$, the reported residuals are 39.7% for state groups and 58.7% for temporal bins, with 11.7 MiB of TSM per task. This is an in-sample comparison for one task and one clustering seed. The storage curve includes all TSM statistics and local maps A_k, whereas the residual calculation subtracts group means and does not apply those maps.

J.12 MAR diagnostic: evaluation data

Figure 4 uses held-out demonstrations from LIBERO-Spatial tasks 0–2. Episodes 0–44 are used for policy training and TSM fitting, and episodes 45–49 are reserved for this diagnostic. Training, evaluation, and codebook clustering use seed 7; the episode split is deterministic. This diagnostic uses 96 codebook clusters per task. The common teacher is the policy after learning tasks 0–2 for 10,000 updates each. Both students start from that teacher and adapt to task 3 for 10,000 updates, using the same data, codebook, and seed, with either unweighted or MAR anchoring.

Evaluation covers every retained frame in the 15 held-out episodes, after removing the final seven frames per episode: 1,652 frames in total, comprising 420, 633, and 599 from tasks 0, 1, and 2. Each frame contributes $H = 16$ action steps and eight independent noise/time draws, giving 211,456 measurements per branch. The branches share the teacher and the query-specific random draws. Draws use the seed rule $\texttt{noise_seed} + 100003j + 1009b + d$, where b is the batch index and d the draw index.

J.13 Transition concentration (panel a)

Frames are grouped into quartiles of teacher pose magnitude ρ_T, with displayed cutoffs 0.65, 0.93, and 1.10 in normalized pose-action coordinates. A frame is marked as near a gripper transition if opening or closing occurs within eight control steps before or after it (± 400 ms). The plotted statistic is the fraction of all marked frames assigned to each quartile: 46%, 20%, 14%, and 21% for Q1–Q4. The dashed line is the 25% uniform reference. The denominator is all near-transition frames, rather than all frames within each quartile. The displayed percentages are rounded. Gripper transitions identify interaction timing, not necessarily successful grasping or release.

J.14 ANGULAR-ALIGNMENT COEFFICIENTS (PANEL B)

Within Q1, Figure 4(b) plots $\rho_{S,h}\rho_{T,h}$ for unweighted anchoring and $w_h\rho_{S,h}\rho_{T,h}$ for MAR. These positive factors are half the magnitudes of the signed cosine coefficients in Equations (10) and (13), respectively. For fixed pose magnitudes and angular error, a larger factor increases the loss penalty for directional disagreement.

The MAR factor uses the final, batch-normalized weight from Equation (12). The two curves use the corresponding students adapted from the common teacher under different anchoring rules. Their difference therefore reflects both the weights and the resulting student magnitudes. Only when student and teacher magnitudes are held fixed does the ratio of weighted to unweighted factors equal w_h. The plotted coefficients characterize angular emphasis in the loss, rather than measured changes in teacher–student angles.

Bins with fewer than 300 measurements are omitted. Bands show the within-bin mean ± 1 standard error for this single-seed diagnostic, rather than variation across independently trained policies.

J.15 HARDWARE EVALUATION

Figure 7 reports directly measured stage training times and peak allocated GPU memory for ER and the task-wise and TPA implementations of TETHER. The hardware evaluation used RTX 4090 and RTX PRO 6000 Blackwell GPUs. Stage timings include TSM construction. General training settings are given in Tables 7 and 8. TPA groups teacher and student queries across past tasks while preserving task-specific weights and loss reductions.

The plotted retained storage excludes the common student policy: TSM uses 11.85 MB per task and one 175.5 MB teacher snapshot, while ER retains 78.4 MB per task. The teacher is replaced between stages. Both TETHER implementations therefore have the same retained storage, while peak GPU memory additionally includes training activations and other allocated tensors.

J.16 Real robot tasks and evaluation

The real robot sequence comprises four manipulation tasks, learned in the order Ring, Box, Carrot, and Cup (Figure 11). Figure 6(b) reports the number of successful rollouts out of ten trials for each evaluated task and stage. Thus, each entry is a success count, rather than a percentage; final-stage average success is computed over the four tasks, with 40 trials in total.

J.16.1 Platform, data, and control

The experiments use an SO follower platform (`so_follower`) with five arm joints and a gripper joint. At 30 Hz, the policy commands six absolute joint target positions: shoulder pan, shoulder lift, elbow flexion, wrist flexion, wrist roll, and gripper position. The gripper command is a continuous absolute position. Two RGB streams recorded under the names `top` and `wrist` are resized from 480×640 to 256×256 for the policy.

We initialize the policy from a model pretrained on three Hugging Face datasets: `roboseasylabs/svla_so101_pickplace`, `lerobot/svla_so100_sorting`, and `lerobot/svla_so100_stacking`. For subsequent lifelong learning, each task supplies 50 demonstrations. Episodes 0–44 are used for training, and episodes 45–49 are held out. ER retains episodes 0–4 of each past task from its training split.

J.16.2 Training and inference

Each of the four tasks is trained for 10,000 updates with batch size 32 and training seed 42, giving 40,000 task updates. ER replay also uses batch size 32. The optimizer is Adam with learning rate 10^{-4}, betas $(0.95, 0.999)$, and weight decay 10^{-6}; the cosine schedule uses 500 warmup updates and the gradient-clipping norm is 10. DINOv2 and CLIP remain frozen. The policy uses two observation steps, predicts 16 action steps with 100 inference steps, and executes eight actions per prediction. The action at index zero corresponds to the preceding observation time and is discarded; indices 1–8 are executed. The held-out demonstrations are separate from the ten live rollouts per task and checkpoint reported in Figure 6(b).

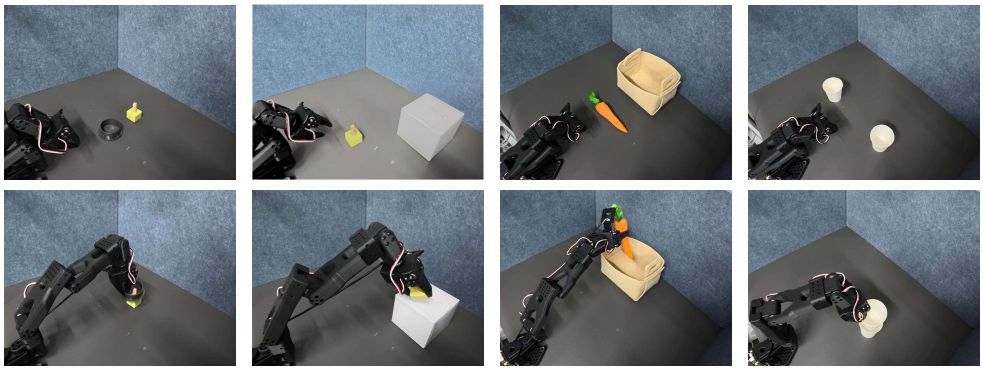


Figure 11: **Real robot task sequence.** Ring placement, block placement, carrot placement, and cup stacking are learned sequentially. The tasks differ in object geometry, grasping constraints, visibility, and placement alignment.

J.16.3 Task 1: Ring

The robot grasps a ring with a wide opening and places it onto the rod of a wooden post. This comparatively simple pick-and-place task begins the sequence and provides a reference for measuring how its performance changes as subsequent tasks are learned.

J.16.4 TASK 2: BOX

The robot grasps a rectangular wooden block from a fixed location and places it on a white box. A ring attached to the block constrains the available grasp. The task requires accurate grasp positioning and controlled placement on the box's smooth surface.

J.16.5 TASK 3: CARROT

The robot grasps a carrot-shaped object from a fixed location and places it in a basket. Its tapered geometry makes grasp placement important. Once grasped, the object obstructs the wrist camera, reducing visual feedback during transport and placement. An unsuccessful grasp can also displace the object and require a corrective attempt.

J.16.6 TASK 4: CUP

The robot grasps the paper cup on the left and inserts it into the cup on the right. The thin cups are susceptible to deformation and slipping, requiring careful grasping and transport. Successful stacking also requires aligning the held cup with the receiving cup before insertion.