IN-CONTEXT IMITATION LEARNING UNDER TASK DRIFT WITH TABULAR FOUNDATION MODELS

Anonymous authors
Paper under double-blind review

ABSTRACT

Behavior cloning is typically formulated as an offline problem: a policy is trained on a fixed demonstration dataset and deployed without further adaptation. In practice, changes in object configurations, initial states, goals, or other task conditions can substantially alter the input distribution and degrade performance. We refer to this setting as task drift and study how a policy can adapt incrementally from small sets of new demonstrations without retraining from scratch. In-context learning provides a natural mechanism for such adaptation without updating model weights. Existing robotic in-context learners, however, acquire this capability through large-scale, robot-specific training, limiting their applicability when demonstrations are scarce and reducing transfer across environments and embodiments. We instead formulate behavioral cloning as tabular in-context regression. A tabular foundation model, pretrained to infer predictive relationships from labeled rows, receives compact feature–action pairs constructed from visual and proprioceptive demonstrations. Accumulating the resulting table provides a memory-efficient policy without training a robot-specific action predictor. We also introduce a fixed-memory alternative to table accumulation, using a learned prompt that is updated through prompt distillation, incorporating the current distribution while preserving the predictive content of the preceding prompt. Our method achieves 88.1% final success, compared with 47.5% for the strongest incremental baseline. It also obtains -3.8 forgetting, indicating backward transfer, while all baselines exhibit substantial forgetting.

1 INTRODUCTION

Behavioral cloning (BC) is one of the most widely used approaches for acquiring robotic behaviors from expert demonstrations. It learns a direct mapping from observations to expert actions through supervised learning. Recent advances have made BC effective across a range of manipulation tasks (Chi et al., 2025). However, this paradigm implicitly assumes that the training demonstrations cover all the conditions that the robot will encounter during deployment. In real-world deployment, a policy may encounter conditions that were not adequately covered by its training demonstrations. For example, a household robot may encounter objects in unfamiliar locations, operate around a rearranged workspace, or approach the same objects from different starting positions. Similarly, a warehouse manipulator may face successive changes in bin locations, item placement, and delivery targets. As a result, a policy that performs well when initially deployed may later perform poorly under the changed conditions. We broadly refer to such changes as *task drift*. Supporting adaptation to such changes is essential for robots expected to operate beyond their initial training conditions.

Adapting to task drift is inherently continual: new scenarios arise throughout deployment, while previously encountered conditions may recur. An expert can demonstrate how to handle an unfamiliar situation, but retraining the robot from scratch after every change would make this interaction impractical. The robot should therefore learn from newly provided demonstrations incrementally without retraining from scratch, while retaining its ability to operate under earlier conditions.

Standard BC stores the learned behavior in the policy parameters. Updating these parameters with new demonstrations can overwrite previously acquired behavior, resulting in catastrophic forgetting (Kirkpatrick et al., 2017). In-context learning provides an alternative in which adaptation occurs through the context while the pretrained model remains fixed. This ability is well established in

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.

large language models, which can infer new tasks from examples supplied at inference time (Brown et al., 2020). Motivated by this property, we consider whether a robot policy can adapt in the same way: new demonstrations specify the required behavior through context, without repeatedly updating or retraining the policy.

Recent work shows that models trained on large collections of robot trajectories can acquire in-context learning capabilities for manipulation (Fu et al., 2025; Vosylius & Johns, 2025; Sridhar et al., 2025). These models, however, are not off-the-shelf imitation learners. Their ability to interpret demonstrations is learned through robot-specific training and is tied to the observation and action interfaces represented in the training data. A model trained for one embodiment therefore cannot be directly applied when the robot, degrees of freedom, sensors, or action dimensions change. We instead formulate behavioral cloning as tabular in-context regression. We use TabPFN (Hollmann et al., 2025), a tabular foundation model pretrained to infer supervised relationships from labeled rows across heterogeneous datasets. Applying it to robot control, however, requires converting demonstrations into a meaningful table. To this end, we learn an action-directed representation that integrates multi-camera visual features, suppresses variation unrelated to control, and combines the resulting features with proprioception. Demonstrations can then be expressed as feature–action rows and supplied directly to TabPFN as context. This allows a pretrained TabPFN to be used as an in-context policy without robot-specific pretraining or updating its weights.

This tabular formulation also provides a practical approach to incremental adaptation. After visual compression, the demonstrations form a compact feature–action table rather than a growing collection of raw images. New distributions can therefore be incorporated by appending their rows to the context. We show that this full-context policy remains memory- and inference-efficient as the table grows. When memory must remain fixed, we replace the growing table with a learned prompt updated incrementally through prompt distillation. Unlike standard prompt tuning, which learns unstructured input embeddings for a fixed downstream dataset, our prompt consists of learnable feature–action rows that preserve the tabular structure interpreted by the frozen predictor. At each step, this prompt is optimized to incorporate the current demonstrations while reproducing the relationships represented by the preceding prompt. This enables constant-memory adaptation without replaying historical trajectories or modifying the pretrained predictor, while preserving earlier behavior. To summarize, our contributions are as follows:

- We introduce incremental task adaptation, an underexplored imitation learning problem in which a policy must adapt to changing task distributions without forgetting earlier behavior.
- We formulate behavioral cloning as tabular in-context learning, representing demonstrations as a table and enabling an off-the-shelf TabPFN model to act as a policy without large-scale robot pretraining or weight updates. To our knowledge, this use of a tabular foundation model for visuomotor control has not been explored previously.
- We introduce two mechanisms for incremental adaptation: a full-context variant that accumulates compact feature–action rows, and a constant-memory prompt-distillation variant that preserves previously learned behavior without replaying historical trajectories.
- We show that our method outperforms strong baselines in the standard offline setting. Under incremental adaptation, our method maintains this advantage while achieving backward transfer, whereas all baselines exhibit severe catastrophic forgetting.

2 Related Work

Behavioral cloning. BC learns an observation-to-action mapping by supervised learning on expert demonstrations and has been widely used in robot manipulation (Hussein et al., 2017; Zare et al., 2024). Considerable research has focused on learning richer conditional action distributions. Energy-based policies, for example, represent actions through an observation-conditioned compatibility function (Florence et al., 2022), while discrete-mode predictors combine action classification with continuous residuals (Shafiullah et al., 2022). Diffusion policies learn an iterative conditional denoising process over actions (Chi et al., 2025). A related direction models behavior over time: conditioning on observation histories or predicting action chunks captures temporal structure and can reduce the number of decisions needed to execute a trajectory (Zhao et al., 2023). These techniques improve how a policy models demonstrations, but do not by themselves specify how knowledge should be maintained across successive deployment distributions.

A complementary direction focuses on generalization and knowledge transfer. Pretrained visual encoders reuse perceptual knowledge from human videos, reducing the number of demonstrations needed to learn downstream control (Nair et al., 2023). Generalist robot policies extend this principle to the action predictor by learning across diverse tasks and robot datasets (O'Neill et al., 2024; Kim et al., 2025; Ghosh et al., 2024; Zitkovich et al., 2023). These approaches broaden transfer beyond task-specific BC through large-scale robot pretraining. Another approach makes the demonstration dataset part of the predictor at inference: retrieval-based BC uses local examples to ground action predictions in observed behavior (Pfeifer et al., 2026). We build on this broader view of demonstration-conditioned prediction by using a pretrained tabular model to infer the feature–action relationship and maintain a learnable context for incremental adaptation.

Continual imitation learning (IL). Continual IL addresses the tension between acquiring new behaviors or skills and retaining previously learned capabilities when demonstrations arrive sequentially. Replay-based approaches mix new data with stored or generated experience to constrain policy updates (Liu et al., 2023); generative replay reconstructs earlier trajectories without retaining the original datasets (Gao et al., 2021). Parameter-regularization approaches restrict changes to important or previously allocated parameters. In pretrained policies, task-specific adapters provide a parameter-efficient way to acquire new tasks while preserving the shared backbone (Liu et al., 2024). Our problem does not learn new behaviors but instead adapts to new settings.

Another family promotes *skill reuse and composition*. The central idea is to represent recurring behaviors as reusable units, allowing new tasks to share prior knowledge rather than learning independent policies. This is implemented through incrementally discovered skill libraries (Wan et al., 2024), retrievable skill adapters (Lee et al., 2024), or reusable primitive prompts (Yao et al., 2025). The emphasis in these manipulation studies is largely on expanding a robot's task or skill repertoire. Our setting instead holds the task fixed within each sequence and changes the geometric conditions under which it must be performed. This corresponds to domain-incremental learning, where the underlying problem remains the same across changing distributions (van de Ven et al., 2022).

Adaptation in imitation learning. Adaptation encompasses several sources of mismatch. Interactive imitation addresses the discrepancy between expert trajectories and the states induced by the learner, obtaining corrective expert labels on the latter (Ross et al., 2011). Related offline approaches synthesize corrective examples near demonstrations by exploiting local continuity of the dynamics (Ke et al., 2024). Few-shot meta-learning and fine-tuning of generalist policies instead use prior experience to make updates from a small demonstration set more effective (Finn et al., 2017; Ghosh et al., 2024; Kim et al., 2025). These approaches concern how to learn under a mismatch, but incremental adaptation additionally requires a mechanism for preserving prior knowledge.

In-context adaptation supplies demonstrations to a policy at inference while keeping its weights fixed (Fu et al., 2025; Vosylius & Johns, 2025; Sridhar et al., 2025). These methods adapt to a target task from a newly supplied context, but they do not consider incremental adaptation in which the context must be repeatedly updated while retaining knowledge of earlier distributions. Moreover, these methods do not provide an in-context predictor that can be directly applied to an arbitrary robot. Applying these methods to a particular robot embodiment requires training with robot trajectories, simulated pseudo-demonstrations, or additional embodiment-specific VLA data. Our in-context predictor requires no such embodiment-specific training. We directly use the predictive prior of a tabular foundation model (Hollmann et al., 2025), which has not been pretrained on robot trajectories, and express demonstrations as a table of feature–action pairs. We further address how a fixed-size context can be updated over repeated shifts without revisiting historical raw demonstrations.

Our approach. We formulate behavioral cloning as in-context regression with a pretrained tabular foundation model. As new demonstrations arrive, they can be directly added to the tabular context, allowing the policy to adapt without modifying the model parameters. We further introduce a bounded-memory alternative that maintains a fixed-size, learnable prompt of synthetic feature–action pairs. At each step, this context is optimized to capture the current demonstrations while preserving the relationships encoded by the previous context. Throughout the task drift sequence, the visual encoder, feature space, and tabular predictor remain fixed. We evaluate both context accumulation and fixed-size prompt adaptation under successive geometric task drifts, measuring adaptation to new conditions and retention of previously encountered ones.

3 PROBLEM DEFINITION

We consider a finite-horizon imitation learning problem defined by a controlled Markov process $\mathcal{M} = (\mathcal{X}, \mathcal{A}, \mathcal{T}, \rho_0)$, where $\mathcal{X}$ is the input space available to the agent, $\mathcal{A}$ is the action space, $\mathcal{T} : \mathcal{X} \times \mathcal{A} \to \mathcal{X}$ denotes the transition dynamics, and ρ_0 is the initial-input distribution. At time step t, the agent receives a low-dimensional robot state $\mathbf{s}_t \in \mathcal{S}$, such as proprioceptive information, and a visual observation $\mathbf{o}_t \in \mathcal{O}$ consisting of images from one or more cameras. The complete input is represented by $\mathbf{x}_t = (\mathbf{s}_t, \mathbf{o}_t) \in \mathcal{X}$. A policy π maps this input to an action $\mathbf{a}_t \in \mathcal{A}$.

An expert demonstration is a trajectory $\tau = \{(\mathbf{x}_t, \mathbf{a}_t^*)\}_{t=0}^{H-1}$, where H is the trajectory horizon and $\mathbf{a}_t^* \in \mathcal{A}$ is the expert action at time t. Initially, the learner receives a dataset of N_0 expert demonstrations, $\mathcal{D}_0 = \{\tau_i^{(0)}\}_{i=1}^{N_0}$. Standard behavioral cloning learns a policy by minimizing the supervised imitation loss:

$$L(\pi) = \mathbb{E}_{(\mathbf{x},\mathbf{a}^*)\sim\mathcal{D}_0} \left[\ell(\pi(\mathbf{x}), \mathbf{a}^*)\right], \tag{1}$$

where ℓ may be a regression loss or the negative log-likelihood of the expert action.

Continual task drift. A policy trained on $\mathcal{D}_0$ may subsequently encounter deployment conditions that are poorly represented by the initial demonstrations. We describe a *task condition* by a latent scene variable $\xi \in \Xi$. This variable can capture any property that affects the observations, state visitation, dynamics, or expert behavior. Examples include changes in object configurations, initial states, goal locations, viewpoints, backgrounds, physical properties, and environmental dynamics.

At adaptation step $k \in \{0, \ldots, K-1\}$, task conditions are drawn from a distribution $p_k(\xi)$, inducing a corresponding distribution over expert trajectories, denoted by $P_k(\tau)$. Task drift occurs when these distributions change over time:

$$P_k(\tau) \neq P_{k-1}(\tau). \tag{2}$$

The semantic task and action interface remain fixed, but the distribution of situations in which the task must be performed changes. Consequently, this setting differs from task-incremental imitation learning (Wan et al., 2024), where each step introduces a distinct skill or task.

At step k, the learner receives a demonstration set $\mathcal{D}_k = \{\tau_i^{(k)}\}_{i=1}^{N_k}$, consisting of N_k expert trajectories from the current deployment distribution P_k. Let π_k denote the policy after incorporating these demonstrations, and let $\mathcal{L}_j(\pi)$ denote the behavioral-cloning loss of policy π under distribution P_j, as defined in Equation 1. The ideal policy at step k minimizes the average loss over all distributions encountered so far:

$$\pi_k^* \in \arg\min_{\pi\in\Pi} \sum_{i=0}^{k} L_i(\pi), \tag{3}$$

where Π denotes the policy class.

Directly optimizing Equation 3 requires retaining demonstrations from every previous distribution and repeatedly retraining the policy on an expanding dataset. The associated storage and computational costs increase throughout deployment, and retraining must be repeated whenever a new distribution is encountered. We instead consider incremental adaptation without replaying historical demonstrations. At step k, the learner has access to the current dataset $\mathcal{D}_k$, but not to the raw datasets $\mathcal{D}_0, \ldots, \mathcal{D}_{k-1}$. The objective is to adapt to P_k while preserving performance on previously encountered distributions, thereby balancing adaptation and retention without retraining the policy on accumulated demonstration data.

4 METHODOLOGY

Existing in-context robot policies typically acquire their capability through large-scale, robot-specific training, tying the resulting predictor to a particular robot embodiment. We avoid this requirement by casting visuomotor imitation as tabular prediction. A frozen visual encoder extracts an embedding from each camera view, after which we learn an action-supervised linear fusion of the views through partial least squares (PLS). This produces a compact low-dimensional representation that retains information relevant to the expert actions. We concatenate this with proprioception to form the policy features $\mathbf{h}_t$; together, $(\mathbf{h}_t, \mathbf{a}_t^*)$ constitutes a row of the policy table. These rows are

accumulated and provided as context to a pretrained tabular foundation model, making incremental adaptation possible without updating the model weights.

As a constant-memory alternative to accumulating the complete feature–action table, we optimize a fixed-size prompt whose feature and action entries remain learnable. During adaptation, this prompt is updated by *prompt distillation*: the new prompt must predict both the newly observed actions and the actions associated with the feature rows in the preceding prompt. This transfers previously acquired behavior across deployment steps while keeping the policy context compact.

Tabular Foundation Models. A tabular foundation model learns a common predictor across many supervised learning problems. A problem is represented by a labeled context

$$\mathcal{C} = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n}, \tag{4}$$

and an unlabeled query $\mathbf{x}$. Given both, the model returns a conditional distribution $p_\theta(y \mid \mathbf{x}, \mathcal{C})$. Since the context specifies the relation to be learned, the same pretrained parameters can solve tables with different feature semantics and target functions.

We use TabPFN, a transformer-based tabular foundation model trained on millions of synthetically generated prediction datasets (Hollmann et al., 2025). Its data generator samples high-level properties of a table using a directed acyclic graph defining a structural causal model. Across pretraining problems, labeled rows are supplied as context and the targets of held-out rows are predicted. In simplified form, the pretrained parameters are obtained from

$$\theta^* = \arg\min_\theta \mathbb{E}_{\mathcal{D}\sim p_{\text{prior}}} \mathbb{E}_{(\mathbf{x},y)\sim\mathcal{D}_{\text{qry}}} \left[-\log p_\theta\left(y \mid \mathbf{x}, \mathcal{D}_{\text{ctx}}\right)\right] \tag{5}$$

This construction gives rise to the term *prior-data fitted network*: the network is fitted to datasets sampled from a prior over data-generating mechanisms and learns to approximate the corresponding posterior predictive rule. Once pretrained, TabPFN performs prediction on a previously unseen table in a forward pass. We exploit this interface for imitation learning by casting the policy as tabular regression. Consequently, a collection of state–action rows is sufficient to instantiate an action predictor; no robot trajectories are used for pretraining.

Constructing the Policy Context. To represent expert demonstrations as a tabular context, the observations from multiple cameras must be integrated into a compact feature vector. Compactness is particularly important because it reduces the memory required to accumulate context rows as more steps are encountered. Let g_ϕ be a frozen pretrained visual encoder and let N_{cam} denote the number of cameras. At time t, we encode and concatenate all views as:

$$\mathbf{v}_t = [g_\phi(\mathbf{o}_t^{(1)}); \ldots; g_\phi(\mathbf{o}_t^{(N_{\text{cam}})})] \in \mathbb{R}^{d_v} \tag{6}$$

The formulation applies to any number of cameras; we use $N_{\text{cam}} = 3$ in our experiments. We seek combinations of these embeddings that explain variation in the expert actions. This lets the demonstrations determine how information from the views should be integrated.

We obtain these combinations with PLS (Wold et al., 2001). Let N denote the total number of state—action pairs across the initial demonstration trajectories. We construct $\mathbf{V} \in \mathbb{R}^{N\times d_v}$ and $\mathbf{A} \in \mathbb{R}^{N\times d_a}$ by stacking the corresponding multi-camera visual embeddings and expert actions. PLS seeks a projection whose latent features have maximum covariance with the expert actions:

$$\mathbf{W}_{\text{PLS}} = \underset{\mathbf{W}^\top\mathbf{W}=\mathbf{I}}{\arg\max} \ \|\text{Cov}(\mathbf{V}\mathbf{W}, \mathbf{A})\|_F^2 , \tag{7}$$

where $\mathbf{W}_{\text{PLS}} \in \mathbb{R}^{d_v\times d_z}$ contains the visual directions retained by PLS. Features caused by factors that do not help explain the demonstrated actions, such as a background that remains unchanged, contribute little to these directions. The result is also low-dimensional, which is useful for the TabPFN model used in our experiments, whose pretraining distribution places practical limits on the supported feature dimension. In practice, this problem is solved iteratively using the NIPALS algorithm, which identifies covariance-maximizing directions. PLS is fitted on the initial demonstrations $\mathcal{D}_0$ and then frozen, ensuring that every incremental step uses the same feature coordinates.

PLS maps the multi-camera embeddings $\mathbf{v}_t$ to a low-dimensional visual representation $\mathbf{z}_t = \mathbf{W}_{\text{PLS}}^\top \mathbf{v}_t$. We concatenate this representation with the proprioceptive information to obtain the state representation $\mathbf{h}_t = [\mathbf{z}_t; \mathbf{s}_t]$. Each expert state–action pair then produces one row of the policy table, $(\mathbf{h}_t, \mathbf{a}_t^*)$. By flattening all expert trajectories, we obtain the complete context $\mathcal{C}_0 = \{(\mathbf{h}_i, \mathbf{a}_i^*)\}_{i=1}^{N}$. Supplying this context to the tabular foundation model defines the policy.

Learning a Compact Policy Prompt. The complete trajectory table can be used directly as context, but it contains many similar state–action pairs and its size grows with the number of demonstrations. We therefore use a compact prompt with a fixed number of rows. Let Φ_k denote the prompt retained after adaptation step k. Each prompt contains M learnable feature–action rows:

$$\Phi_k = \{(\bar{\mathbf{h}}_i, \bar{\mathbf{a}}_i)\}_{i=1}^{M} \tag{8}$$

Initially, each row of Φ_0 is initialized to a state–action pair sampled from the complete context $\mathcal{C}_0$. We then fine-tune this prompt through the frozen TabPFN predictor. This process is analogous to fine-tuning a pretrained model: it minimizes a predictive loss using gradient-based optimization, but applies the updates to the prompt while leaving the pretrained model unchanged. This form of prompt optimization has been shown to adapt prior-fitted networks using a small optimized context (Feuer et al., 2024).

Let $q = \{(\mathbf{h}_i, \mathbf{a}_i)\}_{i=1}^{N_q}$ denote a labeled table of feature–action pairs. Given a prompt Φ, the loss of the pretrained TabPFN regressor f_θ on this query is defined as the negative log-likelihood of the action labels:

$$\mathrm{NLL}(\Phi; q) = -\frac{1}{N_q} \sum_{i=1}^{N_q} \log f_{\theta_0}\left(\mathbf{a}_i \mid \mathbf{h}_i, \Phi\right), \tag{9}$$

Since TabPFN supports only scalar regression targets, we apply this objective separately to each action coordinate and combine the coordinate-wise predictions to form the complete action vector. Because TabPFN is differentiable with respect to its context, we can backpropagate this loss through the frozen predictor and optimize the prompt. At the initial step, the learned prompt is

$$\Phi_0 = \underset{|\Phi|=M}{\arg\min}\ \mathrm{NLL}(\Phi; \mathcal{C}_0) \tag{10}$$

As shown in Section 6, this optimization allows the in-context policy to use a much shorter prompt than the complete trajectory table while retaining its predictive performance. The construction of Φ_k for subsequent steps $k \geq 1$ is described in the incremental prompt-distillation section.

Incremental Prompt Distillation. Incremental prompt optimization using only $\mathcal{C}_k$ can discard behavior learned under earlier distributions. We therefore use the preceding prompt as a source of knowledge distillation. In addition to predicting current demonstrations, the new prompt is required to predict the feature–action rows represented by Φ_{k-1}:

$$\Phi_k = \underset{|\Phi|=M}{\arg\min} \underbrace{\mathrm{NLL}(\Phi; \mathcal{C}_k)}_{\text{prediction loss}} + \lambda \underbrace{\mathrm{NLL}(\Phi; \Phi_{k-1})}_{\text{distillation loss}}, \qquad k \geq 1 \tag{11}$$

where λ controls the adaptation–retention tradeoff. The first term incorporates the new distribution, while the second forces the prompt to correctly predict the feature–action pairs in the previous prompt, preserving its predictive content. Because each previous prompt was constrained by its predecessor, knowledge is carried forward recursively through a memory of constant size. The prompt length remains fixed at (M), historical trajectories are not replayed, and the visual encoder, PLS projection, and TabPFN weights remain unchanged throughout incremental adaptation.

5 Experimental Setup

Tasks and distribution shifts. We construct incremental adaptation sequences from six MetaWorld (Yu et al., 2020) tasks: pick-place, sweep, push, basketball, drawer-close, and button-press-topdown. Each sequence contains $K = 3$ steps, indexed by $k \in 0, \ldots, K-1$, where step 0 corresponds to the original task distribution. Subsequent steps apply geometric shifts to the robot-hand initialization, object position, and goal position regions. We validate that each new step introduces a distinct distribution: policies trained on preceding steps achieve near-zero success when evaluated on the new step. Thus, adaptation requires learning new behavior rather than recovering behavior already represented in the previous demonstrations. Appendix A.1 describes the step construction and validation procedure, including the rationale for using six tasks with three steps within MetaWorld simulator.

For each step, we collect 20 trajectories from a scripted expert. Every trajectory contains proprioceptive states, expert actions, and synchronized RGB observations from three cameras: a corner

view, a top-down view, and a gripper-mounted view. We evaluate two protocols. Under the *joint setting*, a method receives all trajectories collected up to current step. Under *incremental adaptation*, it receives only the current-step trajectories and cannot revisit data from previous steps. At each step, we evaluate the resulting policy on 100 episodes from each distribution encountered so far. We repeat every experiment with three random seeds.

Baselines. We compare against four established imitation-learning methods. **R3M** applies behavioral cloning to pretrained visual encodings using an MLP policy (Nair et al., 2023). **BeT** represents multimodal actions using k-means action tokens and predicts them with a transformer policy (Shafiullah et al., 2022). **ACT** uses a conditional variational autoencoder to predict short action chunks and integrates them through temporal ensembling (Zhao et al., 2023). **DP** predicts actions through a conditional diffusion process (Chi et al., 2025). For incremental adaptation, each policy is trained on step 0 and subsequently fine-tuned using only new demonstrations.

Implementation. We encode each camera view with a frozen ResNet-18 pretrained on robot-manipulation data (Nair et al., 2023), yielding 512-dimensional features for each view. The three views are concatenated and projected by PLS from 1,536 to 32 action-relevant dimensions, then combined with the eight-dimensional proprioceptive state to form a 40-dimensional input paired with a four-dimensional action. We set $\lambda = 1$ and use a default prompt size of $M = 512$, with prompt size studied separately in ablations. Experiments use an NVIDIA A100 GPU and Intel Xeon Gold 6240 CPU. Code is provided in the supplementary material.

Evaluation metrics. We report episode success rate and action RMSE. Success is MetaWorld's binary task-completion signal. Action RMSE compares the policy action with the action returned by the scripted expert at the same state. For each metric, we construct a matrix $A[k, r]$, where k is the training checkpoint and r indexes the evaluation step. Thus, $A[k, r]$ measures performance on the step r distribution after learning k steps. We define the average performance at step k as:

$$\mathrm{A}(k) = \frac{1}{k+1} \sum_{r=0}^{k} A[k, r] \tag{12}$$

We report *Mean* as the average of these step-level scores, $\frac{1}{K} \sum_{k=0}^{K-1} \mathrm{A}(k)$, and *Final* as $\mathrm{A}(K-1)$, which measures the final policy over all distributions. We also measure forgetting by comparing performance on each earlier distribution immediately after it was learned with its performance at the final policy (Chaudhry et al., 2018):

$$\text{Forgetting} = \frac{1}{K-1} \sum_{k=0}^{K-2} \left(A[k, k] - A[K-1, k] \right) \tag{13}$$

For success rate, positive values indicate that final performance is lower than when the distribution was first learned, while negative values indicate backward transfer. Because lower action RMSE is preferable, we reverse the subtraction for RMSE. Thus, positive forgetting consistently indicates degradation for both metrics.

6 Empirical Evaluation

Baseline Comparison. We evaluate two adaptation settings. Under *joint*, each method has access to all demonstrations collected up to the current step. Within this setting, the baselines are retrained on the expanding demonstration dataset, whereas our TabPFN variants construct their context from the accumulated feature–action rows. Under *incremental adaptation*, each method receives only the current-step demonstrations and cannot revisit data from previous steps.

Table 1 shows that our approach outperforms the baselines even when they are allowed joint retraining on all previously observed data. The full-context variant places all accumulated feature–action rows directly in the TabPFN context, without further training, and achieves a final success rate of 93.4%. This exceeds the strongest jointly trained baseline, R3M at 87.1%, by 6.3 percentage points. Prompt tuning further increases the final success rate to 94.2%.

The difference is substantially larger under incremental adaptation. Our prompt-tuned method achieves 88.1% final success, compared with 47.5% for the strongest incremental baseline, a margin of 40.6 points. This gap is explained by the inability of the gradient-trained baselines to retain

Table 1: Aggregate performance across tasks under incremental adaptation and joint settings. Mean measures performance throughout the drift sequence, whereas Final evaluates the last policy over all steps. Negative Forgetting denotes backward transfer to earlier steps. Values are mean $\pm$ standard deviation across seeds after averaging over tasks.

	Success rate (%)			Action RMSE		
Method	Mean ↑	Final ↑	Forgetting ↓	Mean ↓	Final ↓	Forgetting ↓
Incremental adaptation						
R3M	66.9 ± 1.8	47.5 ± 1.4	59.9 ± 0.8	0.549 ± 0.012	0.639 ± 0.017	0.318 ± 0.013
BeT	54.0 ± 0.9	37.2 ± 1.2	62.4 ± 3.3	0.733 ± 0.025	0.837 ± 0.014	0.407 ± 0.040
ACT	45.0 ± 1.7	34.3 ± 0.9	46.7 ± 2.6	0.730 ± 0.018	0.839 ± 0.030	0.418 ± 0.000
DP	35.5 ± 1.0	30.9 ± 1.9	35.7 ± 2.2	0.863 ± 0.006	0.913 ± 0.011	0.310 ± 0.004
Ours (prompt tuning)	$\mathbf{84.0 \pm 2.7}$	$\mathbf{88.1 \pm 2.5}$	$\mathbf{-3.8 \pm 3.6}$	$\mathbf{0.446 \pm 0.002}$	$\mathbf{0.440 \pm 0.008}$	$\mathbf{0.016 \pm 0.006}$
Joint						
R3M	86.3 ± 1.2	87.1 ± 4.0	0.4 ± 3.8	0.453 ± 0.007	0.439 ± 0.009	0.008 ± 0.024
BeT	75.0 ± 1.3	80.1 ± 2.3	-2.7 ± 3.3	0.589 ± 0.002	0.567 ± 0.002	0.006 ± 0.009
ACT	51.7 ± 0.4	57.4 ± 3.1	1.0 ± 7.4	0.666 ± 0.007	0.649 ± 0.033	0.035 ± 0.031
DP	39.6 ± 2.7	43.6 ± 5.5	-3.3 ± 5.6	0.809 ± 0.024	0.778 ± 0.052	-0.024 ± 0.050
Ours (prompt tuning)	$\mathbf{87.6 \pm 0.2}$	$\mathbf{94.2 \pm 0.7}$	-6.6 ± 0.2	0.439 ± 0.008	0.393 ± 0.007	-0.020 ± 0.019
Ours (full context)	84.7 ± 3.2	93.4 ± 0.5	$\mathbf{-9.6 \pm 3.7}$	$\mathbf{0.393 \pm 0.020}$	$\mathbf{0.337 \pm 0.001}$	$\mathbf{-0.032 \pm 0.030}$

earlier distributions when adapting without replay, resulting in 35.7 to 62.4 points of forgetting in success rate. The joint setting largely avoids this problem because every checkpoint is trained on the complete demonstration history, yielding forgetting values close to zero or negative. Our method achieves strong retention: its forgetting value of -3.8 indicates backward transfer, meaning that later prompt updates slightly improve performance on previously encountered distributions. Per-task results for all metrics are reported in Appendix A.2.

Memory and computational cost. The computation and storage required for the joint setting grows with number of steps, as each baseline stores all previous demonstrations and retrains whenever a new distribution is introduced. For example, joint dataset in basketball contains 60 trajectories and 6,212 time steps, occupying 2.81 GB, almost entirely due to the three-camera 224×224 images. Our joint variants discard these images after constructing a $6,212 \times 44$ feature–action table: 32 PLS visual features, eight proprioceptive features, and four action targets. Stored as single precision float values, this table requires only 1.09 MB, over $2,500\times$ less memory than the raw dataset.

The full-context method directly uses this compact table, incorporating each new step by appending its feature–action rows without additional training. The joint prompt-tuned variant instead distills the accumulated table into a fixed prompt of 512 learnable rows through gradient-based optimization while keeping TabPFN frozen. The two variants achieve similar final success rates: 94.2% for prompt tuning and 93.4% for full context. However, full context obtains a lower action RMSE of 0.337 compared to 0.393 for prompt tuning. Despite its growing context, full-context inference remains efficient because TabPFN uses a fixed set of learned inducing points and caches the encoded context, so computation does not grow linearly with context length (Grinsztajn et al., 2026). On basketball, prediction with the complete 6,212-row context takes 184.9 ms and 17.9 GFLOPs, compared with 182.6 ms and 16.8 GFLOPs for the 512-row prompt. Thus, an approximately $12\times$ larger context introduces only a modest inference overhead.

Incremental adaptation follows the standard replay-free continual learning setting and retains neither previous demonstrations nor an accumulated feature–action table. Without access to earlier data, adapting only to the current distribution can overwrite previously acquired behavior. Prompt distillation limits forgetting by fitting the current demonstrations while preserving the relationships encoded by the preceding prompt. This prompt is the method's only persistent memory and occupies 0.090 MB at M=512, regardless of the number of trajectories or steps. Despite having no access to the accumulated table used by joint prompt tuning, the incremental method achieves 88.1% final success, within 6.1 points of the joint result, while exhibiting backward transfer.

Retention across steps. Figure 1 provides a step-by-step view of forgetting. Each entry $A[k, r]$ measures performance on evaluation distribution r after adaptation step k. Reading downward within a column therefore holds the evaluation distribution fixed while advancing the continual learner.

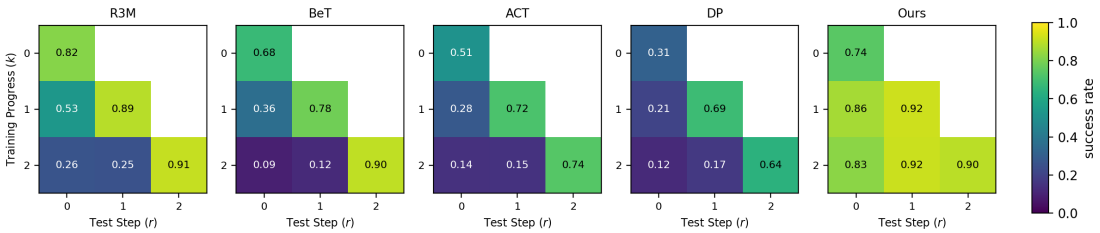

Figure 1: Step-wise success under the incremental adaptation setting. Each entry $A[k, r]$ reports success on step r after training through step k, averaged across all tasks and seeds. Moving downward within a column tracks performance on a fixed step as later steps are learned. The baselines performance on earlier steps degrades sharply, whereas our method maintains or improves its earlier-step performance.

Table 2: Effect of prompt length (M) under incremental adaptation setting, aggregated across all six tasks. Each entry reports the mean $\pm$ standard deviation across seeds.

	Success rate (%)			Action RMSE		
Prompt Length	Mean ↑	Final ↑	Forgetting ↓	Mean ↓	Final ↓	Forgetting ↓
64	64.5 ± 0.0	57.0 ± 2.3	18.1 ± 1.9	0.586 ± 0.023	0.669 ± 0.024	0.136 ± 0.015
128	80.4 ± 3.2	75.6 ± 5.0	12.0 ± 2.3	0.519 ± 0.009	0.533 ± 0.005	0.049 ± 0.011
256	83.8 ± 1.0	85.6 ± 3.2	4.2 ± 4.3	0.480 ± 0.018	0.448 ± 0.000	$\mathbf{0.001 \pm 0.028}$
512	84.0 ± 2.7	88.1 ± 2.5	-3.8 ± 3.6	$\mathbf{0.446 \pm 0.002}$	$\mathbf{0.440 \pm 0.008}$	0.016 ± 0.006
1024	$\mathbf{85.2 \pm 0.2}$	$\mathbf{89.7 \pm 0.9}$	$\mathbf{-5.1 \pm 1.2}$	0.452 ± 0.016	0.450 ± 0.009	0.023 ± 0.013

For every baseline, performance decreases sharply once training progresses beyond the evaluated step. This pattern is most visible in the lower-left region of each heatmap, where success on the initial distribution approaches zero by the final checkpoint. The heatmaps expose the same catastrophic forgetting measured by the aggregate forgetting scores in Table 1. Our method is the only one for which performance remains stable, and often improves, as later steps are learned. Its columns show no systematic decay, consistent with the -3.8 forgetting score and the backward transfer produced by prompt distillation.

Effect of prompt size. Table 2 evaluates the capacity of the learned prompt. Increasing its length from M=64 to M=512 raises final success from 57.0% to 88.1%. The larger prompt can retain more information from earlier distributions while incorporating the current demonstrations. The improvement saturates beyond 512 rows; increasing the prompt to 1,024 rows yields only a further 1.6% improvement, reaching 89.7% at twice the memory cost, while increasing the final action RMSE from 0.440 to 0.450. We therefore use M=512 throughout the main experiments, as it provides a favorable balance between performance and memory.

7 Conclusion

We studied incremental adaptation to task drift, where a robot must learn from demonstrations collected under changing task conditions while retaining earlier acquired behavior. We cast visuomotor behavioral cloning as tabular in-context learning by converting multi-camera observations, proprioceptive states, and expert actions into a compact feature–action table. This formulation enables a pretrained tabular foundation model to serve as a policy without robot-specific pretraining or updates to its weights. For replay-free adaptation, we introduced prompt distillation, which incrementally updates a fixed-size prompt while preserving the relationships represented by its predecessor. Across MetaWorld benchmark, our approach outperforms different imitation learning baselines under both joint and incremental adaptation. In the incremental setting, prompt distillation achieved 88.1% final success, compared with 47.5% for the strongest baseline, while exhibiting backward transfer rather than catastrophic forgetting. These results show that tabular foundation models provide a practical basis for adaptable robot policies, even when large-scale robot-specific pretraining and historical demonstration replay are unavailable.

8 AI USE STATEMENT

In this work, we used generative AI tools to assist with software implementation, experiment execution and analysis, literature review, and manuscript writing and editing. We did not use generative AI for any other tasks requiring disclosure, and the remaining disclosure categories are not applicable to this work. All AI-assisted code was inspected and tested by the authors, and all experimental results were independently checked against the generated outputs. The authors manually reviewed the cited literature, verified the relevance and accuracy of the references, and revised all AI-assisted text for technical correctness, originality, and consistency with the work performed. We take responsibility for the final content of this work, including all text, claims, code, results, and other artifacts produced with the aid of generative AI.

REFERENCES

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*, volume 33, pp. 1877–1901, 2020.

Arslan Chaudhry, Puneet K. Dokania, Thalaiyasingam Ajanthan, and Philip H. S. Torr. Riemannian walk for incremental learning: Understanding forgetting and intransigence. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pp. 556–572, 2018. doi: 10.1007/978-3-030-01252-6_33.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, 44(10–11):1684–1704, 2025. doi: 10.1177/02783649241273668. URL https://journals.sagepub.com/doi/10.1177/02783649241273668.

Benjamin Feuer, Robin T Schirrmeister, Valeriia Cherepanova, Chinmay Hegde, Frank Hutter, Micah Goldblum, Niv Cohen, and Colin White. Tunetables: Context optimization for scalable prior-data fitted networks. *Advances in Neural Information Processing Systems*, 37:83430–83464, 2024.

Chelsea Finn, Tianhe Yu, Tianhao Zhang, Pieter Abbeel, and Sergey Levine. One-shot visual imitation learning via meta-learning. In *Proceedings of the 1st Annual Conference on Robot Learning*, volume 78, pp. 357–368. PMLR, 2017. URL https://proceedings.mlr.press/v78/finn17a.html.

Pete Florence, Corey Lynch, Andy Zeng, Oscar A Ramirez, Ayzaan Wahid, Laura Downs, Adrian Wong, Johnny Lee, Igor Mordatch, and Jonathan Tompson. Implicit behavioral cloning. In *Proceedings of the 5th Conference on Robot Learning*, volume 164, pp. 158–168. PMLR, 2022. URL https://proceedings.mlr.press/v164/florence22a.html.

Letian Fu, Huang Huang, Gaurav Datta, Lawrence Yunliang Chen, William Chung-Ho Panitch, Fangchen Liu, Hui Li, and Ken Goldberg. In-context imitation learning via next-token prediction. In *IEEE International Conference on Robotics and Automation*, 2025. URL https://icrt.dev/.

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

Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman, Charles Xu, Jianlan Luo, You Liang Tan, Lawrence Yunliang Chen, Quan Vuong,

Ted Xiao, Pannag R. Sanketi, Dorsa Sadigh, Chelsea Finn, and Sergey Levine. Octo: An open-source generalist robot policy. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024. doi: 10.15607/RSS.2024.XX.090.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Mihir Manium, Shi Bin Hoo, Magnus Bühler, Anurag Garg, Dominik Safaric, Jake Robertson, Benjamin Jäger, Simone Alessi, Adrian Hayler, Vladyslav Moroshan, Lennart Purucker, Philipp Singer, Alan Arazi, Julien Siems, Jan Hendrik Metzen, Georg Grab, Nick Erickson, Siyuan Guo, Eliott Kalfon, Simon Bing, David Salinas, Clara Cornu, Lilly Charlotte Wehrhahn, Diana Kriuchkova, Kursat Kaya, Lydia Sidhoum, Marie Salmon, Jerry Chen, Madelon Hulsebos, Yann LeCun, Samuel Müller, Bernhard Schölkopf, Sauraj Gambhir, Noah Hollmann, and Frank Hutter. TabPFN-3: Technical report, 2026. URL https://arxiv.org/abs/2605.13986.

Noah Hollmann, Samuel Müller, Lennart Purucker, Arjun Krishnakumar, Max Körfer, Shi Bin Hoo, Robin Tibor Schirrmeister, and Frank Hutter. Accurate predictions on small data with a tabular foundation model. *Nature*, 637:319–326, 2025. doi: 10.1038/s41586-024-08328-6. URL https://www.nature.com/articles/s41586-024-08328-6.

Ahmed Hussein, Mohamed Medhat Gaber, Eyad Elyan, and Chrisina Jayne. Imitation learning: A survey of learning methods. *ACM Computing Surveys (CSUR)*, 50(2):1–35, 2017.

Liyiming Ke, Yunchu Zhang, Abhay Deshpande, Siddhartha Srinivasa, and Abhishek Gupta. CCIL: Continuity-based data augmentation for corrective imitation learning. In *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/99831104028c3b7e6079fd8bdcc42c8f-Abstract-Conference.html.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan P Foster, Pannag R Sanketi, Quan Vuong, Thomas Kollar, Benjamin Burchfiel, Russ Tedrake, Dorsa Sadigh, Sergey Levine, Percy Liang, and Chelsea Finn. OpenVLA: An open-source vision-language-action model. In *Proceedings of the 8th Conference on Robot Learning*, volume 270, pp. 2679–2713. PMLR, 2025. URL https://proceedings.mlr.press/v270/kim25c.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.

Daehee Lee, Minjong Yoo, Woo Kyung Kim, Wonje Choi, and Honguk Woo. Incremental learning of retrievable skills for efficient continual task adaptation. In *Advances in Neural Information Processing Systems*, volume 37, 2024. URL https://papers.nips.cc/paper_files/paper/2024/hash/1f0832859514e53a0e4f229fc9b3a4a2-Abstract-Conference.html.

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*, volume 36, 2023. URL https://proceedings.neurips.cc/paper_files/paper/2023/hash/8c3c666820ea055a77726d66fc7d447f-Abstract-Datasets_and_Benchmarks.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 *International Conference on Learning Representations*, 2024. URL https://proceedings.iclr.cc/paper_files/paper/2024/hash/46e2b525da536030ec1589dddea01feb-Abstract-Conference.html.

Suraj Nair, Aravind Rajeswaran, Vikash Kumar, Chelsea Finn, and Abhinav Gupta. R3M: A universal visual representation for robot manipulation. In *Proceedings of the 6th Conference on Robot Learning*, volume 205, pp. 892–909. PMLR, 2023. URL https://proceedings.mlr.press/v205/nair23a.html.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open X-Embodiment: Robotic learning datasets and RT-X models. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pp. 6892–6903. IEEE, 2024.

Quinn Pfeifer, Ethan Pronovost, Paarth Shah, Khimya Khetarpal, Siddhartha Srinivasa, and Abhishek Gupta. Difference-aware retrieval policies for imitation learning. In *International Conference on Learning Representations*, 2026. URL https://proceedings.iclr.cc/paper_files/paper/2026/hash/8c2ecd140ebe227a531edc9a0c3b69d4-Abstract-Conference.html.

Stephane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics*, volume 15, pp. 627–635. PMLR, 2011. URL https://proceedings.mlr.press/v15/ross11a.html.

Nur Muhammad Mahi Shafiullah, Zichen Jeff Cui, Ariuntuya Altanzaya, and Lerrel Pinto. Behavior transformers: Cloning k modes with one stone. In *Advances in Neural Information Processing Systems*, volume 35, pp. 22955–22968, 2022. doi: 10.52202/068431-1668.

Kaustubh Sridhar, Souradeep Dutta, Dinesh Jayaraman, and Insup Lee. RICL: Adding in-context adaptability to pre-trained vision-language-action models. In *Proceedings of the 9th Conference on Robot Learning*, volume 305, pp. 5022–5038. PMLR, 2025. URL https://proceedings.mlr.press/v305/sridhar25a.html.

Gido M. van de Ven, Tinne Tuytelaars, and Andreas S. Tolias. Three types of incremental learning. *Nature Machine Intelligence*, 4:1185–1197, 2022. doi: 10.1038/s42256-022-00568-3. URL https://www.nature.com/articles/s42256-022-00568-3.

Vitalis Vosylius and Edward Johns. Instant policy: In-context imitation learning via graph diffusion. In *International Conference on Learning Representations*, 2025. URL https://proceedings.iclr.cc/paper_files/paper/2025/hash/5692c7dbc4abcaa50f9ce609819212e5-Abstract-Conference.html.

Weikang Wan, Yifeng Zhu, Rutav Shah, and Yuke Zhu. LOTUS: Continual imitation learning for robot manipulation through unsupervised skill discovery. In *IEEE International Conference on Robotics and Automation*, pp. 537—544, 2024. URL https://ut-austin-rpl.github.io/Lotus/.

Svante Wold, Michael Sjöström, and Lennart Eriksson. PLS-regression: A basic tool of chemometrics. *Chemometrics and Intelligent Laboratory Systems*, 58(2):109–130, 2001. doi: 10.1016/S0169-7439(01)00155-1.

Yuanqi Yao, Siao Liu, Haoming Song, Delin Qu, Qizhi Chen, Yan Ding, Bin Zhao, Zhigang Wang, Xuelong Li, and Dong Wang. Think small, act big: Primitive prompt learning for lifelong robot manipulation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 22573–22583, 2025. URL https://openaccess.thecvf.com/content/CVPR2025/html/Yao_Think_Small_Act_Big_Primitive_Prompt_Learning_for_Lifelong_Robot_CVPR_2025_paper.html.

Tianhe Yu, Deirdre Quillen, Zhanpeng He, Ryan Julian, Karol Hausman, Chelsea Finn, and Sergey Levine. Meta-World: A benchmark and evaluation for multi-task and meta reinforcement learning. In *Proceedings of the Conference on Robot Learning*, volume 100 of *Proceedings of Machine Learning Research*, pp. 1094–1100. PMLR, 2020. URL https://proceedings.mlr.press/v100/yu20a.html.

Maryam Zare, Parham M Kebria, Abbas Khosravi, and Saeid Nahavandi. A survey of imitation learning: Algorithms, recent developments, and challenges. *IEEE Transactions on Cybernetics*, 54(12):7173–7186, 2024.

Tony Z. Zhao, Vikash Kumar, Sergey Levine, and Chelsea Finn. Learning fine-grained bimanual manipulation with low-cost hardware. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023. doi: 10.15607/RSS.2023.XIX.016.

Brianna Zitkovich, Tianhe Yu, Sichun Xu, Peng Xu, Ted Xiao, Fei Xia, Jialin Wu, Paul Wohlhart, Stefan Welker, Ayzaan Wahid, et al. RT-2: Vision-language-action models transfer web knowledge to robotic control. In *Conference on Robot Learning*, pp. 2165–2183. PMLR, 2023.

A Appendix

A.1 Task-Drift Construction

Simulator selection. Generating a valid task drift requires three properties: First, the simulator must expose geometric parameters that affect the task distribution, including robot-hand, object and goal positions. Second, it must provide a procedural script capable of generating demonstrations; simulators that only provide offline datasets cannot be used for task-drift construction. Third, the expert must remain reliable under modified task configurations. Since scripted policies are often designed for a narrow range of object and goal configurations, geometric changes can make the task infeasible for these scripted policies.

We assessed multiple other simulators in this project: CLEVR-Skills, Language-Table, LIBERO, ManiSkill, Ravens, RLBench, RoboCasa, and RoboSuite. None satisfied all three requirements simultaneously. MetaWorld exposes the required geometric parameters, and its scripted policies operate directly on the object, goal, and robot hand positions. These policies consequently tolerate the modified configurations needed to construct new distributions. We therefore use MetaWorld for all experiments.

Task drift construction and validation. A candidate task drift must introduce behavior that is not already covered by the preceding steps. Otherwise, it would lengthen the sequence without creating a meaningful adaptation problem. We test this condition directly during benchmark construction. For each candidate step, we fit our full-context policy on the accumulated demonstrations from all previously accepted steps and evaluate it on the candidate distribution. A candidate is accepted only if this prior policy achieves near-zero success (at most 10%). Candidates already solved by the prior policy are rejected, and a new candidate is sampled. We also require the scripted policy to achieve at least 80% success on the candidate. This ensures that the drift configuration remains feasible for the script and that we can generate expert demonstrations for the new step.

Number of steps. Many MetaWorld tasks do not satisfy the requirements for constructing a valid drift sequence. For some, geometric shifts cause the scripted expert to fail; for others, feasible configurations are already covered by earlier steps. This constraint becomes more restrictive as the sequence grows because the accumulated demonstrations cover an increasing portion of the valid task geometry. Consequently, most tasks cannot satisfy both conditions beyond a third or fourth step. We therefore standardize all drift sequences to K=3 steps, a length that can be constructed consistently across the selected tasks. The resulting benchmark includes pick-place, sweep, push, basketball, drawer-close, and button-press-topdown, covering a diverse range of manipulation skills while providing meaningful distribution shifts.

A.2 Per-Task Results

This section reports the complete per-task results underlying the aggregate evaluation in Table 1. We provide success rate and action RMSE for every task under both incremental adaptation and joint setting.

Table 3: Mean success rate (↑) across MetaWorld tasks. Mean evaluates the performance throughout the drift sequence. Per-task entries are averaged across seeds.

Method	Pick-Place	Sweep	Push	Basketball	Drawer-Close	Button-Press	Average
Incremental adaptation							
R3M	54.6	46.5	64.7	57.2	98.9	79.4	66.9 ± 1.8
BeT	31.4	35.6	52.8	48.5	82.1	73.4	54.0 ± 0.9
ACT	25.4	32.2	51.3	14.8	94.6	51.7	45.0 ± 1.7
DP	31.8	9.3	37.9	0.6	94.6	39.0	35.5 ± 1.0
Ours (prompt tuning)	74.5	68.1	85.7	75.4	100.0	100.0	$\mathbf{84.0 \pm 2.7}$
Joint							
R3M	82.4	62.0	87.4	86.3	100.0	100.0	86.3 ± 1.2
BeT	61.1	35.9	80.5	73.2	99.6	99.6	75.0 ± 1.3
ACT	33.2	39.2	46.9	11.1	100.0	79.8	51.7 ± 0.4
DP	22.9	6.8	33.4	1.3	99.8	73.4	39.6 ± 2.7
Ours (prompt tuning)	75.2	78.9	86.7	84.6	100.0	100.0	$\mathbf{87.6 \pm 0.2}$
Ours (full context)	59.0	90.2	87.8	87.9	100.0	83.3	84.7 ± 3.2

Table 4: Final success rate (↑) across MetaWorld tasks. Final evaluates the last policy over all steps. Per-task entries are averaged across seeds.

Method	Pick-Place	Sweep	Push	Basketball	Drawer-Close	Button-Press	Average
Incremental adaptation							
R3M	40.2	17.6	55.2	37.3	96.8	38.1	47.5 ± 1.4
BeT	28.7	25.7	40.2	38.9	56.2	33.3	37.2 ± 1.2
ACT	12.7	23.0	33.0	20.0	83.8	33.3	34.3 ± 0.9
DP	31.8	7.9	36.7	0.4	83.8	24.9	30.9 ± 1.9
Ours (prompt tuning)	85.1	68.6	94.7	80.3	100.0	100.0	$\mathbf{88.1 \pm 2.5}$
Joint							
R3M	91.6	50.3	96.4	84.0	100.0	100.0	87.1 ± 4.0
BeT	77.2	33.7	90.3	80.8	99.6	98.8	80.1 ± 2.3
ACT	24.5	60.7	60.0	8.2	100.0	90.8	57.4 ± 3.1
DP	15.9	6.8	46.9	1.9	99.8	90.4	43.6 ± 5.5
Ours (prompt tuning)	89.8	81.5	97.2	96.8	100.0	100.0	$\mathbf{94.2 \pm 0.7}$
Ours (full context)	75.0	94.0	96.8	94.8	100.0	100.0	93.4 ± 0.5

Table 5: Success Forgetting (↓) across MetaWorld tasks. Forgetting measures the decrease in performance on earlier steps between the checkpoint at which they were learned and the final policy. Per-task entries are averaged across seeds.

Method	Pick-Place	Sweep	Push	Basketball	Drawer-Close	Button-Press	Average
Incremental adaptation							
R3M	71.2	66.3	49.8	74.3	4.8	92.8	59.9 ± 0.8
BeT	52.3	48.0	61.2	48.7	65.0	99.3	62.4 ± 3.3
ACT	52.2	35.2	72.5	15.0	24.2	81.0	46.7 ± 2.6
DP	55.0	14.3	49.5	0.3	24.3	70.5	35.7 ± 2.2
Ours (prompt tuning)	-5.0	-8.2	-7.3	-2.5	0.0	0.0	$\mathbf{-3.8 \pm 3.6}$
Joint							
R3M	-6.0	17.2	-8.7	-0.0	0.0	0.0	0.4 ± 3.8
BeT	-7.0	8.2	-9.2	-9.2	0.0	1.2	-2.7 ± 3.3
ACT	29.5	-24.3	-10.5	12.5	0.0	-1.0	1.0 ± 7.4
DP	16.5	-3.3	-17.2	-0.3	-0.2	-15.5	-3.3 ± 5.6
Ours (prompt tuning)	-7.5	-10.5	-9.8	-11.8	0.0	0.0	-6.6 ± 0.2
Ours (full context)	-9.2	-7.0	-8.0	-8.5	0.0	-25.0	$\mathbf{-9.6 \pm 3.7}$

Table 6: Mean Action RMSE (↓) across MetaWorld tasks. Mean evaluates the performance throughout the drift sequence. Per-task entries are averaged across seeds.

Method	Pick-Place	Sweep	Push	Basketball	Drawer-Close	Button-Press	Average
Incremental adaptation							
R3M	0.521	0.726	0.569	0.670	0.442	0.365	0.549 ± 0.012
BeT	0.767	0.906	0.687	0.869	0.552	0.616	0.733 ± 0.025
ACT	0.703	0.752	0.713	1.050	0.437	0.727	0.730 ± 0.018
DP	0.772	1.032	0.773	1.006	0.562	1.031	0.863 ± 0.006
Ours (prompt tuning)	0.399	0.560	0.584	0.598	0.305	0.229	$\mathbf{0.446 \pm 0.002}$
Joint							
R3M	0.407	0.636	0.457	0.602	0.409	0.204	0.453 ± 0.007
BeT	0.533	0.922	0.525	0.716	0.485	0.353	0.589 ± 0.002
ACT	0.605	0.737	0.727	1.032	0.364	0.529	0.666 ± 0.007
DP	0.643	1.073	0.657	1.033	0.534	0.918	0.809 ± 0.024
Ours (prompt tuning)	0.436	0.599	0.445	0.570	0.353	0.230	0.439 ± 0.008
Ours (full context)	0.379	0.446	0.419	0.616	0.211	0.285	$\mathbf{0.393 \pm 0.020}$

Table 7: Final Action RMSE (↓) across MetaWorld tasks. Final evaluates the last policy over all steps. Per-task entries are averaged across seeds.

Method	Pick-Place	Sweep	Push	Basketball	Drawer-Close	Button-Press	Average
Incremental adaptation							
R3M	0.588	0.801	0.662	0.740	0.434	0.607	0.639 ± 0.017
BeT	0.915	0.903	0.808	0.845	0.570	0.981	0.837 ± 0.014
ACT	0.910	0.723	0.859	1.040	0.536	0.966	0.839 ± 0.030
DP	0.884	1.015	0.882	0.998	0.555	1.144	0.913 ± 0.011
Ours (prompt tuning)	0.338	0.530	0.620	0.636	0.261	0.257	$\mathbf{0.440 \pm 0.008}$
Joint							
R3M	0.369	0.676	0.367	0.609	0.395	0.219	0.439 ± 0.009
BeT	0.467	0.896	0.497	0.650	0.535	0.359	0.567 ± 0.002
ACT	0.724	0.622	0.621	1.005	0.408	0.517	0.649 ± 0.033
DP	0.620	1.062	0.570	1.121	0.392	0.904	0.778 ± 0.052
Ours (prompt tuning)	0.380	0.533	0.342	0.562	0.341	0.201	0.393 ± 0.007
Ours (full context)	0.310	0.429	0.302	0.566	0.251	0.166	$\mathbf{0.337 \pm 0.001}$

Table 8: Action RMSE Forgetting (↓) across MetaWorld tasks. Forgetting measures the decrease in performance on earlier steps between the checkpoint at which they were learned and the final policy. Per-task entries are averaged across seeds.

Method	Pick-Place	Sweep	Push	Basketball	Drawer-Close	Button-Press	Average
Incremental adaptation							
R3M	0.326	0.283	0.414	0.191	0.121	0.576	0.318 ± 0.013
BeT	0.615	0.152	0.511	0.165	0.172	0.826	0.407 ± 0.040
ACT	0.588	0.089	0.543	0.168	0.369	0.749	0.418 ± 0.000
DP	0.545	0.150	0.451	0.142	0.202	0.372	0.310 ± 0.004
Ours (prompt tuning)	−0.029	−0.025	0.140	0.031	−0.075	0.057	$\mathbf{0.016 \pm 0.006}$
Joint							
R3M	−0.035	0.104	−0.088	0.042	0.013	0.009	0.008 ± 0.024
BeT	−0.015	0.008	0.039	−0.052	0.089	−0.031	0.006 ± 0.009
ACT	0.284	−0.120	−0.095	0.028	0.073	0.042	0.035 ± 0.031
DP	−0.004	−0.020	−0.077	0.162	−0.203	−0.001	-0.024 ± 0.050
Ours (prompt tuning)	0.008	−0.068	−0.073	0.020	0.026	−0.033	-0.020 ± 0.019
Ours (full context)	−0.023	0.032	−0.094	−0.013	0.095	−0.190	$\mathbf{-0.032 \pm 0.030}$