GRIP: A GENERALIST IN-CONTEXT REWARD MODEL FOR ROBOTIC MANIPULATION

Anonymous authors
Paper under double-blind review

ABSTRACT

Dense rewards provide step-level feedback for assessing progress, selecting demonstrations, and training policies, yet specifying them separately for every task is costly. We present GRIP, an in-context reward model that predicts a task's dense reward from a small set of labeled observations supplied at inference time. Its main configuration combines a frozen visual encoder with a frozen in-context regressor, allowing tasks to change through context rather than parameter updates. On MetaWorld and ManiSkill, GRIP improves dense-reward prediction over regressors fitted on the same examples. Its predictions improve with more context, depend on the observation–reward pairing, and remain stable when a shared context contains up to 50 tasks. On ProcVQA, a lightweight adapter trained on global visual features raises VOC from 0.6287 to 0.8133. GRIP's predicted rewards also select demonstrations that improve behavior-cloning success. Together, these results show how context-specified rewards can support progress assessment and data selection across robotic manipulation and embodied procedural tasks.

1 INTRODUCTION

Dense rewards describe progress at each step of a manipulation task, providing feedback for policy learning, trajectory evaluation, and demonstration selection. Obtaining these rewards across tasks is costly: hand-designed rewards require repeated tuning, simulation rewards depend on privileged state, and human feedback requires annotation effort (Christiano et al., 2017; Ma et al., 2023b; Zhang et al., 2025; Lee et al., 2021). Learned reward models reduce manual design by acquiring visual representations and reward functions from videos, robot data, or process annotations (Ma et al., 2023b; Zhang et al., 2025; Feng et al., 2026; Lee et al., 2026). These approaches provide reusable knowledge, but judging progress in a new task still requires a way to specify what counts as progress. As robot datasets expand to cover more goals and environments (O'Neill et al., 2024; Khazatsky et al., 2024), we seek a model that can obtain this specification from a small amount of task-specific evidence.

A few observations paired with numerical rewards can provide that evidence directly. Consider a drawer that is almost closed: the same image may indicate high progress for closing the drawer and low progress for opening it. For either task, labeled examples of different drawer states show which visual changes matter and how they map to the reward scale. A model can then use these examples as references when evaluating a new observation. This suggests specifying the reward function through a labeled context at inference time. The resulting problem is in-context regression: given observation–reward pairs and a query image, predict the query's numerical reward without fitting parameters to the target task.

Solving this problem requires both a representation of the relevant visual differences and a way to infer their relationship to reward from examples. We bring these capabilities together in GRIP, a generalist in-context reward model (Figure 1). A frozen visual encoder maps context and query images into the same feature space. A frozen in-context regressor, pretrained on synthetic regression tasks, takes the context feature–reward pairs and the query feature as input and returns a reward prediction in a forward pass. The visual representation is shared across tasks, while the supplied labels determine how it should be interpreted for the current task. This training-free configuration requires task labels but no additional reward-model training.

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

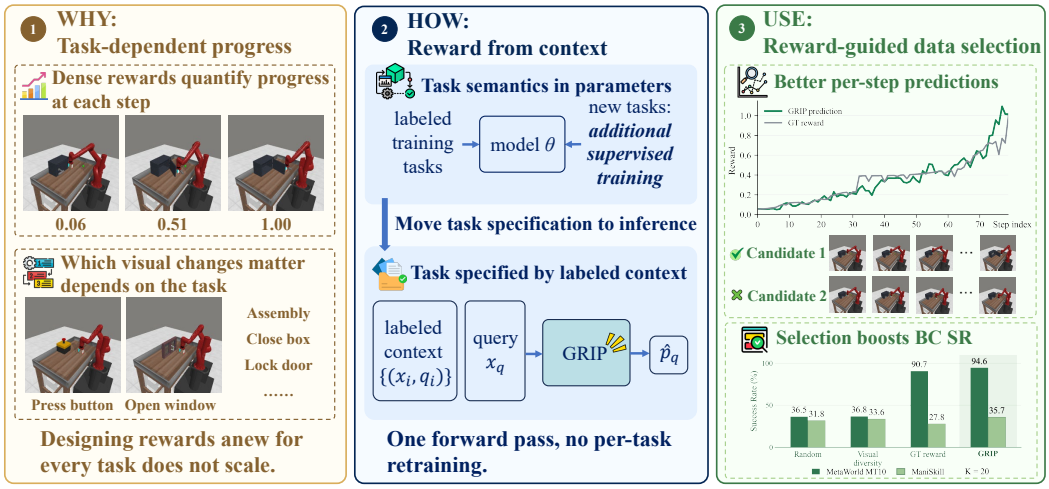

Figure 1: Why and how GRIP uses context to specify rewards. Left: progress depends on the task, making dense rewards costly to define separately. Center: labeled examples specify the task at inference without retraining. Right: predicted progress guides demonstration selection; the bars show BC success on MetaWorld MT10 and ManiSkill with $K = 20$ selected trajectories per task.

Because task information enters through examples, the same model can evaluate a different task by changing its context. It can also handle several tasks in a shared context: each image feature is concatenated with a frozen task-text feature that identifies which task its reward belongs to, and the query includes the corresponding task description. Each task retains its own labeled examples. The usefulness of these examples depends on the visual information available to the regressor. We therefore also study a lightweight trained adapter that transforms visual features, examining when feature learning improves on training-free inference.

We evaluate reward prediction on MetaWorld and ManiSkill using separate context and query trajectories. With the same visual features and 100 labeled context points per task, training-free GRIP outperforms four fitted regressors in both reward ranking and numerical accuracy. Controlled experiments show how the context contributes: increasing its size improves prediction, disrupting image–reward pairings reduces accuracy, and combining up to fifty MetaWorld tasks preserves performance when each task supplies 100 labeled examples. We also evaluate procedural progress on ProcVQA, where training-free GRIP trails ProcVLM. A trained adapter improves prediction from the same global features. A separate robot study adds patch tokens as well as adapter training and finds only small gains on held-out tasks.

Beyond prediction accuracy, we test whether the estimated rewards preserve information useful for learning policies. GRIP scores candidate demonstrations by their predicted progress near the end of each trajectory, then combines these scores with visual diversity to select a training set. A separate behavior-cloning policy learns from the selected demonstrations. On both robot benchmarks, this selection procedure yields higher success rates than random or diversity-only selection. This experiment tests the practical value of predicted rewards by selecting useful demonstrations without accessing the candidate trajectories' environment rewards.

Our contributions are:

- **Training-free, context-specified rewards.** GRIP combines a frozen visual encoder with a frozen in-context regressor to predict dense rewards from labeled observations. New tasks are specified through context without task-specific parameter updates.
- **Reward prediction and context analysis.** Comparisons on two manipulation benchmarks evaluate ranking and numerical accuracy, while controlled experiments examine context size, label pairing, and multi-task inference. Separate adapter studies investigate the role of learned features, including on ProcVQA.
- **Reward-guided demonstration selection.** Combining predicted progress with visual diversity selects demonstrations that improve behavior-cloning success on both robot benchmarks.

2 Related Work

Visual rewards and progress. Visual reward methods learn from preferences, temporal video structure, goal images, or language (Christiano et al., 2017; Ma et al., 2023b;a; Ayalew et al., 2025; Zhang et al., 2025). Language descriptions and demonstrations can also specify visual rewards (Rocamonde et al., 2024; Sontakke et al., 2023). These sources supply useful visual semantics, but a goal image or language instruction alone need not determine the numerical reward at every intermediate state. Process-aware and generalist vision-language models instead learn progress judgments from large annotated corpora (Feng et al., 2026; Lee et al., 2026; Tan et al., 2026). GRIP uses progress-labeled observations supplied at inference to define both the task-specific mapping and its reward scale. The frozen visual representation supplies features; labeled context determines how those features should be interpreted for the current task.

In-context value and reward inference. The value model V_0 (Zhang et al., 2026a) joins a pretrained encoder and in-context regressor with a trained adapter, but predicts a policy's initial-state value. Vision-language approaches can infer per-frame value from examples or derive rewards from token probabilities (Ma et al., 2025; Chen et al., 2026). GRIP instead predicts numerical per-transition rewards from visual features and labeled context. The main robot configuration uses no adapter training; optional feature adaptation is examined separately. This distinction matters because changing the regressor's context and updating a visual representation are different mechanisms of transfer.

Demonstration selection. Prior work selects imitation data using quality heuristics, success labels, or visual diversity (Belkhale et al., 2023; Zhang et al., 2026b; Mirzasoleiman et al., 2020; Kuhar et al., 2023; Chen et al., 2025; Agia et al., 2025). Quality and diversity serve different purposes: a high-scoring trajectory may be redundant with one already selected, while a diverse trajectory may make little progress. GRIP combines predicted progress with visual coverage to rank candidate demonstrations without their environment rewards. This provides a downstream test of whether predicted rewards preserve information useful for policy training.

3 GRIP

GRIP predicts a query observation's dense reward from labeled visual examples. Its training-free configuration combines a frozen visual encoder with a frozen in-context regressor: the encoder represents observations, and the regressor infers their relationship to reward from context. This configuration underlies the main robot prediction, context-analysis, and demonstration-selection experiments. We first describe prediction for a single task, then feature adaptation, shared contexts, and demonstration selection.

3.1 Problem and In-Context Reward Prediction

For a task τ, let o_t denote the image observed after transition t and $p_t \in [0, 1]$ its normalized dense-reward label. We use progress to denote this task-specific reward label, without assuming that it increases monotonically along a trajectory. Given a labeled context $\mathcal{C}_\tau = \{(o_i, p_i)\}_{i=1}^{M}$ and a query image o_q, the goal is to estimate the query's reward under τ. The index i enumerates context samples, which may come from different trajectories and cover different stages and outcomes, including failures. Their image–reward pairings specify the visual changes relevant to the task and the numerical scale of the prediction.

Visual representation and regression. We use a frozen CLIP ViT-B/16 (Radford et al., 2021) image encoder f and L2-normalize its 512-dimensional global embedding:

$$x(o) = \frac{f(o)}{\|f(o)\|_2}. \tag{1}$$

Applying the same encoder and normalization to context and query images places them in a common feature space. The post-transition observation aligns each visual input with the reward assigned to that transition. The predictor receives image features and context labels, but no robot actions or privileged simulator state.

Table 1: GRIP configurations and their visual inputs. Training-free GRIP passes global features directly to the regressor. The trained adapter uses global features on ProcVQA and global features plus patch tokens in the robot ID/OOD study. Only adapter parameters are trained; the encoder and regressor remain frozen.

Configuration	Visual input	Evaluation
Training-free	Global feature	Robot main results; ProcVQA baseline
Trained adapter	Global feature	ProcVQA trained variant
Trained adapter	Global + patch tokens	Robot ID/OOD comparison

We use TabPFN 2.5 (Grinsztajn et al., 2025) as the frozen in-context regressor g. Its pretraining on synthetic regression tasks enables prediction from a table of feature–label examples. Each context observation contributes a row containing its image feature and reward; the query supplies its feature without a reward label. The prediction is

$$\hat{p}_q = g\big(\{(x(o_i), p_i)\}_{i=1}^{M},\, x(o_q)\big)\,. \tag{2}$$

The forward pass conditions the pretrained regressor on the supplied examples without updating its parameters or fitting a separate predictor for the task.

The encoder provides reusable visual features, while the paired labels tell the regressor how those features relate to reward for the current task. The same visual state can have different rewards under different goals, as in opening and closing a drawer. Context labels specify both the direction of progress and its numerical scale. Changing the labels therefore changes the regression problem even if the images remain fixed; our context interventions test whether predictions follow this change. In the robot evaluation protocol, context and query trajectories are disjoint, and query labels are reserved for evaluation.

Optional feature adaptation. A lightweight adapter learns a reward-relevant transformation of the frozen visual features. Let $h_\phi(o)$ denote the feature supplied to g. Training-free inference uses $h_\phi(o) = x(o)$ and has no trainable ϕ. An adapted configuration applies a feature map a_ϕ to the encoder's global feature, optionally together with its patch tokens. Context and query observations pass through the same map before regression:

$$\hat{p}_q = g\big(\{(h_\phi(o_i), p_i)\}_{i=1}^{M},\, h_\phi(o_q)\big)\,. \tag{3}$$

The adapter is trained using the query reward-prediction loss. In the robot adapter experiment, each step samples one training task, 100 labeled context points, and 16 query points. Context labels enter the regressor as conditioning information, while query labels provide the supervision for its predictions. Gradients propagate through g to update the adapter parameters ϕ; the encoder and regressor parameters remain fixed. At evaluation, the adapter is fixed as well, and task adaptation occurs through the labeled context. Training therefore learns a feature map shared across tasks, while inference uses the current task's examples to predict its rewards.

Table 1 distinguishes the evaluated configurations. The ProcVQA adapter uses global features only, whereas the robot ID/OOD adapter also receives patch tokens. The robot comparison thus changes both the visual input and the training status, preventing attribution of its differences to training alone. Both adapter studies are separate from the training-free robot experiments.

3.2 Multi-Task Joint Inference

When several tasks share a context, each observation's reward must be associated with its task. We attach a task description to every context point and to the query, and encode it with the frozen CLIP text encoder f_{text} to obtain z. Joint inference uses the frozen global image features, concatenated with the corresponding text features:

$$\hat{p}_q = g\big(\{([x(o_i); z_i],\, p_i)\}_{i=1}^{M},\, [x(o_q); z_q]\big)\,. \tag{4}$$

Here M is the total number of context points across tasks, while z_i and z_q identify the tasks of context point i and the query. Text features identify the task, while numerical labels specify its reward mapping. In a single-task context, all examples already share a task, so no description is

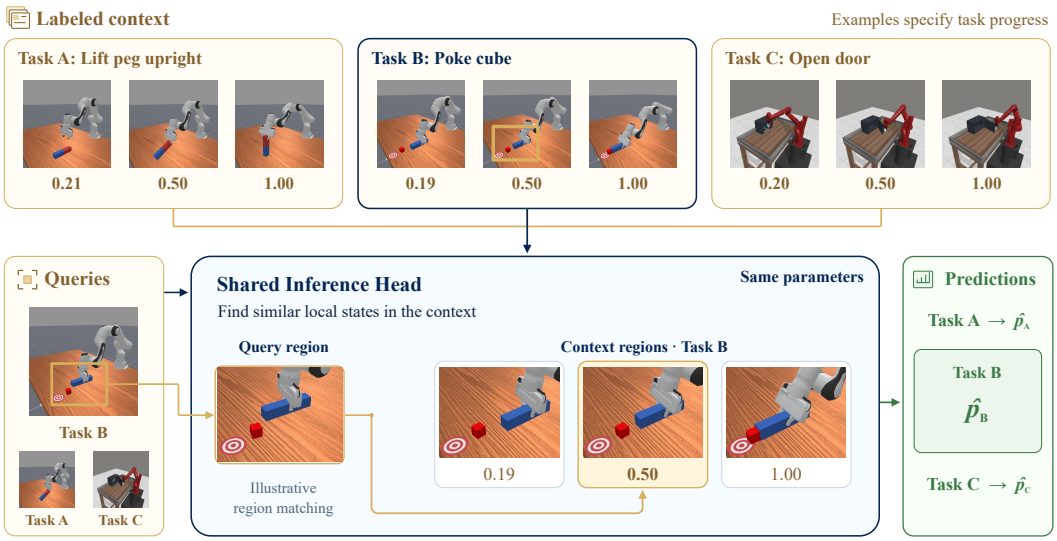

Figure 2: Reward prediction from labeled context. Examples of lifting a peg, poking a cube, and opening a door specify different progress mappings for the same inference head. Query predictions use the corresponding task context without changing model parameters. Highlighted regions illustrate visual correspondences; the training-free implementation uses global image features, not explicit region matching.

needed. Adding a task requires its labeled observations and description, with no parameter update. In our MetaWorld experiment, each task contributes 100 points, so expanding from one to fifty tasks grows the total context from 100 to 5,000 points. Every query task is represented in the context. This protocol tests interference from other tasks as the shared context grows.

3.3 Reward-Guided Demonstration Selection

We select K demonstrations for behavior cloning by combining predicted reward with visual diversity. For a candidate trajectory $\xi = (o_1, \ldots, o_L)$, we first predict the reward $\hat{p}_t$ at each transition. Its quality score is the β-quantile of predictions over the final fraction ρ of the trajectory, with $\rho, \beta \in (0, 1)$:

$$s(\xi) = \text{Quantile}_\beta\left(\{\hat{p}_t\}_{t=L-h+1}^{L}\right), \qquad h = \max(1, \lceil \rho L \rceil). \tag{5}$$

The tail window emphasizes progress near the end, and choosing a low quantile reduces the influence of isolated high predictions. We use $\rho = 0.1$ and $\beta = 0.2$, so the score is the 20th percentile of predicted rewards over the final 10% of transitions.

To combine quality with coverage, let $r(\xi) \in [0, 1]$ denote the percentile rank of $s(\xi)$ in the candidate pool, with higher values indicating higher scores. Each trajectory is represented by the image feature of its first post-transition frame. After per-dimension standardization, $d(\xi, \mathcal{S})$ measures the nearest distance from this feature to those of trajectories in the selected set $\mathcal{S}$. We initialize $\mathcal{S}$ with the highest-scoring trajectory, then repeatedly add the remaining candidate maximizing

$$\alpha\, r(\xi) + (1 - \alpha)\, \frac{d(\xi, \mathcal{S})}{\max_{\xi' \notin \mathcal{S}} d(\xi', \mathcal{S})}, \tag{6}$$

until K trajectories are selected. The weight $\alpha \in [0, 1]$ balances quality and diversity. The reward ranks are fixed within the candidate pool; distances and the maximum over remaining candidates are recomputed as $\mathcal{S}$ grows. Thus each new selection accounts for the visual configurations already covered, reducing redundancy among high-scoring demonstrations.

The reward-model context and candidate trajectories come from disjoint source groups, so related versions of a trajectory cannot appear in both. Candidates are scored using predicted rewards without accessing their environment rewards. A separate BC policy is then trained on the selected trajec-

Table 2: Robot reward prediction on 50 MetaWorld and eight ManiSkill tasks. Training-free GRIP and the four fitted regressors share frozen CLIP features and 100 labeled context points; Qwen uses 50 images with aligned task coverage. Query trajectories are disjoint from context. Values are macro means over tasks and three context seeds; arrows indicate the preferred direction.

	MetaWorld			ManiSkill		
Method	Spearman ↑	C-index ↑	NRMSE ↓	Spearman ↑	C-index ↑	NRMSE ↓
kNN	0.6187	0.7356	0.1022	0.8087	0.8721	0.1795
Ridge	0.4620	0.6720	0.1088	0.8135	0.8746	0.1714
GP	0.4063	0.6472	0.1057	0.2535	0.6090	0.3040
XGBoost	0.5033	0.6881	0.1010	0.7861	0.8593	0.1894
Qwen3.7 Plus	0.5844	0.7002	0.1069	0.8168	0.8813	0.1704
Qwen3.8 Max	0.6215	0.7371	0.1045	0.8050	0.8672	0.2106
GRIP (ours)	**0.6811**	**0.7663**	**0.0924**	**0.8276**	**0.8906**	**0.1697**

tories, with states and task text as inputs and recorded actions as targets. Predicted rewards only determine which demonstrations enter training; selection leaves the reward model unchanged.

4 EXPERIMENTS

We first test whether training-free GRIP predicts rewards accurately, how those predictions depend on context, and whether they help select demonstrations for behavior cloning. We then evaluate a trained adapter to determine how learning a feature map affects prediction with the encoder and regressor held fixed.

Shared setup. Training-free GRIP combines L2-normalized 512-dimensional CLIP ViT-B/16 global features (Radford et al., 2021) with frozen TabPFN 2.5 (Grinsztajn et al., 2025). We evaluate robot reward prediction on all 50 MetaWorld MT50 tasks and eight ManiSkill v3 tasks. MetaWorld trajectories are generated by scripted policies with varying action noise and include successful runs and failures at different progress levels. ManiSkill uses official demonstration replays, retaining both successes and failures.

Each image is labeled with the normalized environment reward of the preceding transition. Per task, we sample ten uniformly spaced frames from each of ten trajectories to form a 100-point context. Query trajectories are disjoint; MetaWorld also separates context and query task variants. We report Spearman and C-index for ranking and NRMSE for error normalized by the query-label range, macro-averaging task-level metrics over three context seeds. Dataset construction and metric definitions are in Appendix A.

4.1 TRAINING-FREE REWARD PREDICTION

Robot benchmarks. We compare GRIP with kNN, Ridge, GP, and XGBoost using the same frozen features and context/query split. These baselines fit only on context labels, allowing us to compare regression methods without changing the visual representation. Query labels are used only for evaluation.

Qwen3.7 Plus and Qwen3.8 Max provide multimodal-model comparisons. Their input content and task coverage align with GRIP, but they use 50 context images instead of 100. This budget performed best when we tested 20%, 50%, 80%, and 100% of GRIP's context budget on a subset.

GRIP obtains the best reported values on all three metrics in both benchmarks (Table 2). Its Spearman reaches 0.6811 on MetaWorld and 0.8276 on ManiSkill, exceeding kNN by 0.0624 and 0.0189. It also exceeds the stronger Qwen model on each benchmark, Qwen3.8 Max on MetaWorld and Qwen3.7 Plus on ManiSkill, by 0.0596 and 0.0108. The advantage is smaller on ManiSkill, where Qwen3.7 Plus's NRMSE of 0.1704 is close to GRIP's 0.1697. The comparisons favor GRIP across both suites, though the size of the improvement depends on the benchmark.

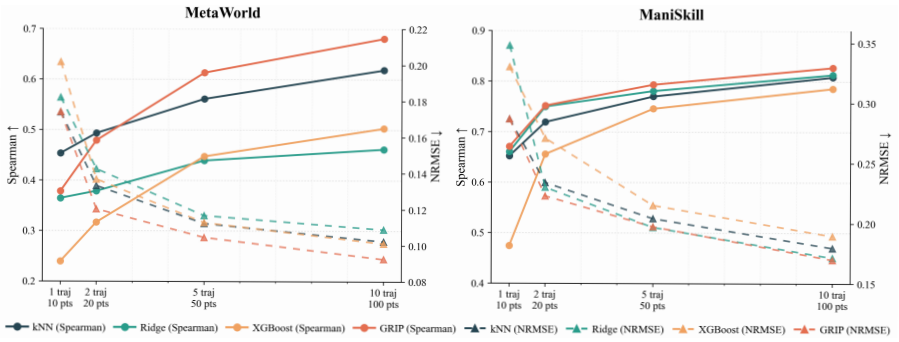

Figure 3: Effect of context size on MetaWorld (left) and ManiSkill (right). Each trajectory supplies 10 labeled points. Colors distinguish GRIP, kNN, Ridge, and XGBoost; solid lines with circles show Spearman on the left axis, and dashed lines with triangles show NRMSE on the right axis. More context improves both GRIP metrics without parameter updates.

ProcVQA. We also evaluate training-free GRIP on ProcVQA to test procedural progress in a different visual domain. We follow ProcVLM's VOC and EPR@50 definitions (Feng et al., 2026) and use 100 context images, with at least one task from the query's category represented. GRIP obtains 0.6287 VOC and 4.9504 EPR@50, exceeding GPT-5.4 but trailing ProcVLM. This gap motivates examining whether a learned feature map improves prediction from the same visual inputs.

4.2 How Training-Free GRIP Uses Context

Context size. We first vary how many labeled examples the model receives. Increasing context from 10 to 100 points raises Spearman from 0.3787 to 0.6811 on MetaWorld and from 0.6708 to 0.8276 on ManiSkill (Figure 3). NRMSE falls from 0.1740 to 0.0924 and from 0.2867 to 0.1697, respectively. More context improves both ordering and numerical accuracy without parameter updates.

Context mapping interventions. To distinguish the effect of correct labels from simply providing more images, we hold context size at 100 points and disrupt its correspondence between images and rewards. We shuffle labels, substitute another task's context, or retain the images while resampling labels from the task's marginal distribution.

Shuffling reduces Spearman from 0.6819 to 0.0259 on MetaWorld and from 0.8266 to 0.0285 on ManiSkill, with C-index near chance (Table 3). Context from another task retains weak correlation, while marginal resampling reduces it below zero. These results show that prediction depends on which reward is paired with each image, beyond the images and reward distribution alone.

Multi-task context. We next test whether examples from other tasks interfere with prediction. Joint contexts contain 1, 2, 5, 10, 20, or 50 MetaWorld tasks, each contributing 100 labeled points and CLIP task-text features. Spearman stays near 0.67–0.68 and NRMSE ranges from 0.0906 to 0.0943 (Figure 4). This stability holds as total context grows to 5,000 points. Every query task is represented, so the result concerns interference among included tasks; it does not establish performance at a fixed total budget or on tasks absent from context.

4.3 Selecting Demonstrations with Training-Free GRIP

We evaluate reward-guided demonstration selection on MetaWorld MT10 and eight ManiSkill tasks. Training-free GRIP selects $K \in \{20, 40, 60\}$ of 320 candidate trajectories using predicted progress and visual diversity (Equations (5) and (6)). We compare random selection, diversity-only selection, and selection using environment rewards (GT). BC data are separate from the reward-prediction benchmark. The 100 labeled context points and 320 candidate trajectories come from disjoint source groups, preventing overlap between related trajectory variants.

Table 3: Context interventions for training-free GRIP with 100 labeled points. Shuffled permutes labels; Wrong task substitutes another task's context; Random marginal retains images but resamples labels from the task's reward distribution. Values are task and context-seed macro means. N/A denotes undefined Spearman for constant predictions; C-index 0.5 indicates chance ordering.

	MetaWorld			ManiSkill		
Setting	Spearman ↑	C-index ↑	NRMSE ↓	Spearman ↑	C-index ↑	NRMSE ↓
Correct	**0.6819**	**0.7667**	**0.0922**	**0.8266**	**0.8901**	**0.1700**
Shuffled	0.0259	0.5096	0.1337	0.0285	0.5085	0.3524
Wrong task	0.2296	0.5803	0.2562	0.2045	0.5908	0.4719
Random marginal	-0.0436	0.4849	0.1322	-0.0086	0.4954	0.3510
Constant	N/A	0.5000	0.1348	N/A	0.5000	0.3484

Table 4: BC success (%) after selecting K of 320 trajectories per task. GRIP uses predicted progress and visual diversity; GT uses environment rewards for the same trajectory tail score. Selectors share training and evaluation settings within each seed. Values are means ± sample standard deviations over three paired seeds. All uses every candidate and has no K budget.

	MetaWorld MT10			ManiSkill		
Selection method	K=20	K=40	K=60	K=20	K=40	K=60
Random	36.5 ± 9.2	48.6 ± 7.0	39.5 ± 11.6	31.8 ± 2.3	37.2 ± 1.1	38.9 ± 1.9
Visual diversity	36.8 ± 2.6	43.9 ± 10.6	34.9 ± 15.7	33.6 ± 6.5	39.6 ± 1.3	41.2 ± 3.8
GT reward	90.7 ± 1.7	94.9 ± 1.9	97.8 ± 1.2	27.8 ± 1.2	36.3 ± 2.1	41.5 ± 0.4
GRIP	**94.6 ± 3.6**	**95.7 ± 1.5**	**96.2 ± 0.2**	**35.7 ± 3.4**	**40.4 ± 3.1**	**44.1 ± 3.5**
All	53.4 ± 14.2 (all 320; no selection)			41.6 ± 3.6 (all 320; no selection)		

Predicted rewards determine selection only. The BC policy takes states and task text as inputs and learns to predict recorded actions, with equal total loss weight per trajectory regardless of length. Across three paired seeds, selectors share candidate pools, policy initialization, update budgets, and evaluation episodes.

At $K = 20$, GRIP yields 94.6% MetaWorld and 35.7% ManiSkill success, versus 36.5% and 31.8% for random selection (Table 4). It outperforms random and diversity-only selection at every budget, with larger gains on MetaWorld. At $K = 60$, GRIP reaches 96.2% and 44.1%, compared with GT's 97.8% and 41.5%.

Using all 320 candidates gives lower success than GRIP at $K = 60$, even with longer training matched to the $K = 40$ data exposure (Appendix C). Extra updates do not close the gap, although selection changes both data quality and quantity.

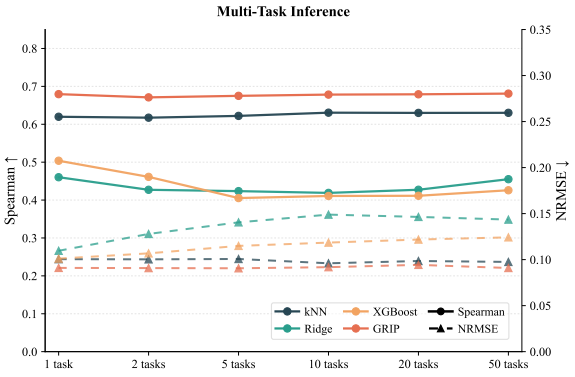

Figure 4: Reward prediction with 1 to 50 MetaWorld tasks in a shared context. Each task contributes 100 labeled points, growing the total from 100 to 5,000; all query tasks are represented. Colors identify methods. Solid lines with circles show Spearman (left axis), and dashed lines with triangles show NRMSE (right axis).

4.4 Reward Prediction with a Trained Adapter

The preceding experiments use context to predict rewards without training the reward model, but the ProcVQA results leave room for improvement. We now evaluate GRIP with a trained adapter, which learns a feature map while keeping the encoder and regressor frozen. Adapter weights remain fixed during evaluation, and context still specifies the task. We first keep the global-feature input unchanged on ProcVQA, then evaluate a robot configuration that also receives patch tokens.

Table 5: Procedural progress estimation on ProcVQA. VOC and EPR@50 follow ProcVLM; higher is better. Bold marks the best value per column. Both GRIP configurations use 100 context images, the same frozen CLIP global features, and frozen TabPFN. Only GRIP with a trained adapter learns a feature map; neither uses patch tokens.

Method	VOC ↑	EPR@50 ↑
GPT-5.4	0.5229	4.8058
ProcVLM	0.8058	**5.7489**
GRIP (Training-free)	0.6287	4.9504
GRIP (Trained adapter)	**0.8133**	5.7026

Table 6: Reward prediction on training tasks (ID) and held-out test tasks (OOD), with 100 context points. Values are means ± sample standard deviations across tasks and context seeds. Bold marks the best reported mean within each split. The trained adapter also adds patch tokens. †ManiSkill ID compares different query pools, so its differences are not matched training gains.

		MetaWorld		ManiSkill	
Split	Configuration	Spearman ↑	NRMSE ↓	Spearman ↑	NRMSE ↓
ID	Training-free	0.6622 ± 0.1577	0.0898 ± 0.0346	$0.7907 \pm 0.1430^{\dagger}$	$0.1718 \pm 0.0858^{\dagger}$
ID	Trained adapter	$\mathbf{0.7588} \pm 0.1571$	$\mathbf{0.0783} \pm 0.0336$	$\mathbf{0.8711} \pm 0.0943$	$\mathbf{0.1144} \pm 0.0688$
OOD	Training-free	0.6980 ± 0.1728	0.0946 ± 0.0403	0.8351 ± 0.0171	0.1539 ± 0.0362
OOD	Trained adapter	$\mathbf{0.7127} \pm 0.2057$	$\mathbf{0.0924} \pm 0.0424$	$\mathbf{0.8357} \pm 0.0227$	$\mathbf{0.1523} \pm 0.0366$

ProcVQA. Both ProcVQA configurations use the same global features, context budget, and frozen regressor, with no patch tokens. Training the adapter raises VOC from 0.6287 to 0.8133 and EPR@50 from 4.9504 to 5.7026 (Table 5). This comparison shows the benefit of learning a feature map over the existing representation. The trained adapter exceeds ProcVLM on VOC but remains below it on EPR@50.

Robot task generalization. The robot study asks whether a trained adapter also helps on tasks excluded from training. Here the adapter receives global features and patch tokens, following V_0's dynamic-query architecture (Zhang et al., 2026a). Global features generate offsets to learned queries, which aggregate visual tokens through cross-attention. Each step samples one training task, 100 context points, and 16 queries. The query prediction loss propagates through frozen TabPFN to update only the adapter, using AdamW with learning rate 10^{-4} and weight decay 0.01.

MetaWorld uses 20 training, 5 validation, and 25 test tasks; ManiSkill uses 3, 2, and 3. We evaluate the step-10,000 and step-16,000 checkpoints, respectively, with 100 context points and three context seeds per task. ID evaluates training tasks, while OOD evaluates test tasks only. For ManiSkill ID, the trained adapter uses a 20% episode holdout but the training-free baseline uses the original complete pool, so their scores are not directly matched. Further details are in Appendix A.3.

The trained adapter has higher ID Spearman and lower ID NRMSE (Table 6), subject to the ManiSkill split difference above. On OOD tasks, Spearman changes from 0.6980 to 0.7127 on MetaWorld and from 0.8351 to 0.8357 on ManiSkill. These gains and the NRMSE changes are small relative to variation across tasks and seeds. Unlike the ProcVQA comparison, the robot study changes both training status and visual input, so it cannot attribute the gains to adapter training alone.

5 Conclusion and Limitations

We presented GRIP, a visual reward model that uses labeled context to specify tasks without updating its frozen encoder or regressor. On MetaWorld and ManiSkill, it outperforms regressors fitted to the same context and selects effective demonstrations for behavior cloning. Predictions improve with more examples and rely on correct image–reward pairings. A trained global-feature adapter improves ProcVQA performance, while robot adapter gains on held-out tasks remain small and do not isolate training from the addition of patch tokens. GRIP still requires task-specific progress labels and depends on what its frozen encoder can extract from single images, which may be insufficient for temporally ambiguous tasks. Real-robot validation remains open.

AI USE STATEMENT

Generative AI tools assisted with manuscript revision and figure preparation. For the implementation, their role was limited to checking code correctness and reviewing engineering details; the core implementation was developed by the authors.

REFERENCES

Christopher Agia, Rohan Sinha, Jingyun Yang, Rika Antonova, Marco Pavone, Haruki Nishimura, Masha Itkina, and Jeannette Bohg. CUPID: Curating Data your Robot Loves with Influence Functions. In *Conference on Robot Learning*, pp. 2907–2932, 2025.

Tewodros W. Ayalew, Xiao Zhang, Kevin Yuanbo Wu, Tianchong Jiang, Michael Maire, and Matthew R. Walter. PROGRESSOR: A Perceptually Guided Reward Estimator with Self-Supervised Online Refinement. In *International Conference on Computer Vision*, pp. 10297–10306, 2025.

Suneel Belkhale, Yuchen Cui, and Dorsa Sadigh. Data quality in imitation learning. In *Advances in Neural Information Processing Systems*, volume 36, 2023.

Jingjing Chen, Hongjie Fang, Hao-Shu Fang, and Cewu Lu. Towards effective utilization of mixed-quality demonstrations in robotic manipulation via segment-level selection and optimization. In *IEEE International Conference on Robotics and Automation*, pp. 16884–16891, 2025.

Shirui Chen, Cole Harrison, Ying-Chun Lee, Angela Jin Yang, Zhongzheng Ren, Lillian J. Ratliff, Jiafei Duan, Dieter Fox, and Ranjay Krishna. TOPReward: Token Probabilities as Hidden Zero-Shot Rewards for Robotics. *arXiv preprint arXiv:2602.19313*, 2026.

Paul F. Christiano, Jan Leike, Tom Brown, Miljan Martic, Shane Legg, and Dario Amodei. Deep reinforcement learning from human preferences. In *Advances in Neural Information Processing Systems*, volume 30, 2017.

Youhe Feng, Hansen Shi, Haoyang Li, Xinlei Guo, Yang Wang, Chengyang Zhang, Jinkai Zhang, Xiaohan Zhang, Jie Tang, and Jing Zhang. ProcVLM: Learning Procedure-Grounded Progress Rewards for Robotic Manipulation. *arXiv preprint arXiv:2605.08774*, 2026.

Léo Grinsztajn, Klemens Flöge, Oscar Key, Felix Birkel, Philipp Jund, Brendan Roof, Benjamin Jäger, Dominik Safaric, Simone Alessi, Adrian Hayler, et al. TabPFN-2.5: Advancing the State of the Art in Tabular Foundation Models. *arXiv preprint arXiv:2511.08667*, 2025.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, Peter David Fagan, Joey Hejna, Masha Itkina, Marion Lepert, Yecheng Jason Ma, Patrick Tree Miller, Jimmy Wu, Suneel Belkhale, Shivin Dass, Huy Ha, Arhan Jain, Abraham Lee, Youngwoon Lee, Marius Memmel, Sungjae Park, Ilija Radosavovic, Kaiyuan Wang, Albert Zhan, Kevin Black, Cheng Chi, Kyle Beltran Hatch, Shan Lin, Jingpei Lu, Jean Mercat, Abdul Rehman, Pannag R. Sanketi, Archit Sharma, Cody Simpson, Quan Vuong, Homer Rich Walke, Blake Wulfe, Ted Xiao, Jonathan Heewon Yang, Arefeh Yavary, Tony Z. Zhao, Christopher Agia, Rohan Baijal, Mateo Guaman Castro, Daphne Chen, Qiuyu Chen, Trinity Chung, Jaimyn Drake, Ethan Paul Foster, Jensen Gao, David Antonio Herrera, Minho Heo, Kyle Hsu, Jiaheng Hu, Donovon Jackson, Charlotte Le, Yunshuang Li, Roy Lin, Zehan Ma, Abhiram Maddukuri, Suvir Mirchandani, Daniel Morton, Tony Nguyen, Abigail O'Neill, Rosario Scalise, Derick Seale, Victor Son, Stephen Tian, Emi Tran, Andrew E. Wang, Yilin Wu, Annie Xie, Jingyun Yang, Patrick Yin, Yunchu Zhang, Osbert Bastani, Glen Berseth, Jeannette Bohg, Ken Goldberg, Abhinav Gupta, Abhishek Gupta, Dinesh Jayaraman, Joseph J. Lim, Jitendra Malik, Roberto Martín-Martín, Subramanian Ramamoorthy, Dorsa Sadigh, Shuran Song, Jiajun Wu, Michael C. Yip, Yuke Zhu, Thomas Kollar, Sergey Levine, and Chelsea Finn. DROID: A large-scale in-the-wild robot manipulation dataset. In *Robotics: Science and Systems*, 2024.

Sachit Kuhar, Shuo Cheng, Shivang Chopra, Matthew Bronars, and Danfei Xu. Learning to discern: Imitating heterogeneous human demonstrations with preference and representation learning. In *Conference on Robot Learning*, pp. 1437–1449, 2023.

Kimin Lee, Laura M. Smith, and Pieter Abbeel. PEBBLE: Feedback-efficient interactive reinforcement learning via relabeling experience and unsupervised pre-training. In *International Conference on Machine Learning*, pp. 6152–6163, 2021.

Tony Lee, Andrew Wagenmaker, Karl Pertsch, Percy Liang, Sergey Levine, and Chelsea Finn. RoboReward: General-Purpose Vision-Language Reward Models for Robotics. *arXiv preprint arXiv:2601.00675*, 2026.

Yecheng Jason Ma, Vikash Kumar, Amy Zhang, Osbert Bastani, and Dinesh Jayaraman. LIV: Language-Image Representations and Rewards for Robotic Control. In *International Conference on Machine Learning*, pp. 23301–23320, 2023a.

Yecheng Jason Ma, Shagun Sodhani, Dinesh Jayaraman, Osbert Bastani, Vikash Kumar, and Amy Zhang. VIP: Towards Universal Visual Reward and Representation via Value-Implicit Pre-Training. In *International Conference on Learning Representations*, 2023b.

Yecheng Jason Ma, Joey Hejna, Chuyuan Fu, Dhruv Shah, Jacky Liang, Zhuo Xu, Sean Kirmani, Peng Xu, Danny Driess, Ted Xiao, et al. Vision language models are in-context value learners. In *International Conference on Learning Representations*, 2025.

Baharan Mirzasoleiman, Jeff Bilmes, and Jure Leskovec. Coresets for data-efficient training of machine learning models. In *International Conference on Machine Learning*, pp. 6950–6960, 2020.

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 *IEEE International Conference on Robotics and Automation*, pp. 6892–6903, 2024.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pp. 8748–8763, 2021.

Juan Rocamonde, Victoriano Montesinos, Elvis Nava, Ethan Perez, and David Lindner. Vision-language models are zero-shot reward models for reinforcement learning. In *International Conference on Learning Representations*, 2024.

Sumedh Sontakke, Jesse Zhang, Séb Arnold, Karl Pertsch, Erdem Bıyık, Dorsa Sadigh, Chelsea Finn, and Laurent Itti. RoboCLIP: One demonstration is enough to learn robot policies. In *Advances in Neural Information Processing Systems*, volume 36, pp. 55681–55693, 2023.

Huajie Tan, Sixiang Chen, Yijie Xu, Zixiao Wang, Cheng Chi, Yuheng Ji, Yaoxu Lyu, Zhongxia Zhao, Xiansheng Chen, Peterson Co, Shaoxuan Xie, Guocai Yao, Pengwei Wang, Zhongyuan Wang, and Shanghang Zhang. General process reward modeling for robotic reinforcement learning. In *Conference on Computer Vision and Pattern Recognition*, pp. 22412–22422, 2026.

Jiahui Zhang, Yusen Luo, Abrar Anwar, Sumedh Anand Sontakke, Joseph J. Lim, Jesse Thomason, Erdem Biyik, and Jesse Zhang. ReWiND: Language-Guided Rewards Teach Robot Policies without New Demonstrations. In *Conference on Robot Learning*, 2025.

Yi-Kai Zhang, Zhiyuan Yao, Hongyan Hao, Yueqing Sun, Qi Gu, Hui Su, Xunliang Cai, De-Chuan Zhan, and Han-Jia Ye. V_0: A Generalist Value Model for Any Policy at State Zero. In *International Conference on Machine Learning*, 2026a.

Yu Zhang, Yuqi Xie, Huihan Liu, Rutav Shah, Michael Wan, Linxi Fan, and Yuke Zhu. SCIZOR: A Self-Supervised Approach to Data Curation for Large-Scale Imitation Learning. In *IEEE International Conference on Robotics and Automation*, 2026b.

A Additional Experimental Details

A.1 Benchmark Construction and Data Splits

MetaWorld reward prediction. MetaWorld MT50 covers 50 tasks. Scripted policies with varying action noise generate successful trajectories and high-, medium-, and low-progress failures. Per task, ten variants form the context pool and ten disjoint variants form the query pool. Four trajectories per variant give 40 trajectories in each pool: 4,000 trajectories and 515,368 transitions overall. Each post-transition image is labeled with its normalized environment dense reward.

ManiSkill reward prediction. We replay official demonstrations for LiftPegUpright, PegInsertionSide, PickCube, PokeCube, PullCube, PushCube, RollBall, and StackCube to obtain images and normalized dense rewards. Per task, 100 trajectories form a fixed context pool; the rest form the query set, totaling 6,959 trajectories and 279,876 transitions. Both successful and failed replays are retained, without quality-stratum balancing during context sampling.

Context sampling. We sample 10 uniformly spaced labeled transitions from each of 1, 2, 5, or 10 trajectories per task, giving 10, 20, 50, or 100 context points. All context experiments use three seeds and disjoint context/query trajectories; main comparisons and pairing interventions use 100 points. Multi-task contexts contain 1, 2, 5, 10, 20, or 50 MetaWorld tasks, each contributing 100 points; every query task is represented.

A.2 Implementation and Metrics

Frozen features and comparators. GRIP and the fitted regressors share L2-normalized 512-dimensional CLIP ViT-B/16 post-transition image features; GRIP uses frozen TabPFN 2.5. Baselines fit only on context labels. kNN uses cosine distance; its neighbor count and weighting, and Ridge's regularization, are chosen by trajectory-grouped cross-validation (up to five folds) within the context. Both models are then refitted on all context samples. GP fits a $\mathrm{ConstantKernel} \times \mathrm{RBF} + \mathrm{WhiteKernel}$ kernel. XGBoost uses 100 trees, depth 2, and learning rate 0.05. Query labels are excluded from fitting and selection. Multi-task inputs concatenate CLIP task-text and image features.

Qwen comparison. Qwen3.7 Plus and Qwen3.8 Max use input content and task coverage aligned with GRIP, but 50 context images instead of 100. On a subset, we tested 20%, 50%, 80%, and 100% of GRIP's context-image budget; 50% performed best and was used for evaluation.

Metric definitions. For a task–seed query set Q with labels p_j and predictions $\hat{p}_j$, normalized RMSE is

$$\mathrm{NRMSE}(Q) = \frac{\sqrt{|Q|^{-1} \sum_{j \in Q} (\hat{p}_j - p_j)^2}}{\max_{j \in Q} p_j - \min_{j \in Q} p_j}. \tag{7}$$

Spearman measures rank agreement. C-index averages over unordered query pairs with unequal true labels, assigning 1 to correct ordering, 0.5 to a predicted tie, and 0 to reversed ordering. Metrics are computed per task–seed query set and macro-averaged; variation is the sample standard deviation across these units, not the standard error of the mean. Constant predictions have undefined Spearman and C-index 0.5.

A.3 Adapter Training and ID/OOD Evaluation

Models and optimization. The training-free baseline passes CLIP global features directly to TabPFN 2.5. The trained adapter aggregates frozen patch tokens and global features using V_0's dynamic-query design (Zhang et al., 2026a). A two-layer MLP (linear, LayerNorm, GELU, dropout, linear) maps the global feature to offsets added to learned attention queries; its final layer is zero-initialized. Cross-attention uses layer-normalized queries and keys; keys and values come from linearly projected tokens. Flattened query outputs feed TabPFN. We use 8 attention queries of dimension 64, 4 attention heads, a 128-dimensional MLP bottleneck, and dropout 0.2.

Table 7: Robot reward prediction with 100 labeled context points, including variability for the five methods that share frozen CLIP features. Values are means ± sample standard deviations across tasks and three context seeds. GRIP uses the training-free configuration; each baseline fits only on context labels.

Method	MetaWorld Spearman ↑	MetaWorld C-index ↑	MetaWorld NRMSE ↓	ManiSkill Spearman ↑	ManiSkill C-index ↑	ManiSkill NRMSE ↓
kNN	0.6187 ± 0.1679	0.7356 ± 0.0728	0.1022 ± 0.0403	0.8087 ± 0.1086	0.8721 ± 0.0517	0.1795 ± 0.0502
Ridge	0.4620 ± 0.1721	0.6720 ± 0.0750	0.1088 ± 0.0366	0.8135 ± 0.0798	0.8746 ± 0.0384	0.1714 ± 0.0424
GP	0.4063 ± 0.2222	0.6472 ± 0.0968	0.1057 ± 0.0371	0.2535 ± 0.3614	0.6090 ± 0.1737	0.3040 ± 0.0780
XGBoost	0.5033 ± 0.1713	0.6881 ± 0.0727	0.1010 ± 0.0334	0.7861 ± 0.0935	0.8593 ± 0.0447	0.1894 ± 0.0502
GRIP	$\mathbf{0.6811 \pm 0.1625}$	$\mathbf{0.7663 \pm 0.0739}$	$\mathbf{0.0924 \pm 0.0395}$	$\mathbf{0.8276 \pm 0.0928}$	$\mathbf{0.8906 \pm 0.0444}$	$\mathbf{0.1697 \pm 0.0579}$

Only the adapter is trained through TabPFN's reward-prediction loss, using AdamW (learning rate 10^{-4}, weight decay 0.01) for 20,000 steps. Each step samples one training task, 100 context points (ten trajectories, ten points each), and 16 query observations. Training and evaluation use normalized rewards.

Task splits and evaluation. MetaWorld uses a 20/5/25 train/validation/test task split (seed 0) and the step-10,000 checkpoint. ManiSkill uses a 3/2/3 split (seed 3), the step-16,000 checkpoint, and a 20% episode holdout from training tasks for ID evaluation. Each task uses three context seeds, 100 context points, and the full corresponding query pool. OOD results exclude validation tasks. Table 6 reports task–seed means and sample standard deviations ($\mathrm{ddof} = 1$): MetaWorld has 20×3 ID and 25×3 OOD records; ManiSkill has 3×3 in each split.

Training also changes the input representation. MetaWorld ID evaluates training tasks' query pools, not held-out tasks. ManiSkill's frozen ID baseline uses the original complete trajectory pool, whereas its trained variant uses the 20% episode holdout; a matching frozen baseline is unavailable. OOD differences are small relative to task–seed variation. ProcVQA has no patch-token ID/OOD evaluation.

A.4 ProcVQA Global-Feature Adapter

VOC and EPR@50 follow ProcVLM (Feng et al., 2026). GRIP uses 100 context images, matching the main experiment's budget, with at least one task from the query's category represented.

Both ProcVQA variants use frozen 512-dimensional CLIP global features and frozen TabPFN. The trained variant optimizes an adapter through the reward-prediction loss without patch tokens. Its input and evaluation protocol differ from the robot adapter study.

B Additional Dense-Reward Results

Table 7 gives means and standard deviations for GRIP and the four fitted regressors.

B.1 Context Budget and Pairing

Table 8 gives results across context budgets.

Table 9 reports pairing interventions. Shuffled permutes context labels; Wrong task substitutes another task's samples; Random marginal retains images and resamples labels from the task's marginal distribution. The interventions change the information supplied in context while leaving the model fixed; they do not identify which visual cues the model uses.

B.2 Multi-Task Context

Table 10 reports multi-task results with 100 labeled points per task.

Table 8: Training-free GRIP with 10, 20, 50, or 100 labeled context points, sampled as 10 points from each of 1, 2, 5, or 10 trajectories. Query trajectories are disjoint. Values are means ± sample standard deviations across tasks and three context seeds; model parameters remain fixed across budgets.

Benchmark	Context points	Spearman ↑	C-index ↑	NRMSE ↓
MetaWorld	10	0.3787 ± 0.2321	0.6401 ± 0.0924	0.1740 ± 0.0692
	20	0.4793 ± 0.2167	0.6785 ± 0.0875	0.1201 ± 0.0531
	50	0.6134 ± 0.1761	0.7336 ± 0.0758	0.1044 ± 0.0403
	100	0.6811 ± 0.1625	0.7663 ± 0.0739	0.0924 ± 0.0395
ManiSkill	10	0.6708 ± 0.1466	0.7946 ± 0.0702	0.2867 ± 0.1091
	20	0.7520 ± 0.0961	0.8379 ± 0.0466	0.2226 ± 0.0644
	50	0.7937 ± 0.0956	0.8669 ± 0.0471	0.1972 ± 0.0700
	100	0.8276 ± 0.0928	0.8906 ± 0.0444	0.1697 ± 0.0579

Table 9: Full results for the 100-point context interventions in Table 3. Values are means ± sample standard deviations across tasks and three context seeds. Shuffled and Random marginal retain context images but change labels; Wrong task replaces the context. Constant predictions have undefined Spearman and C-index 0.5.

	MetaWorld			ManiSkill		
Setting	Spearman ↑	C-index ↑	NRMSE ↓	Spearman ↑	C-index ↑	NRMSE ↓
Correct	0.6819 ± 0.1618	0.7667 ± 0.0736	0.0922 ± 0.0391	0.8266 ± 0.0946	0.8901 ± 0.0452	0.1700 ± 0.0593
Shuffled	0.0259 ± 0.2416	0.5096 ± 0.0858	0.1337 ± 0.0490	0.0285 ± 0.2272	0.5085 ± 0.0939	0.3524 ± 0.0566
Wrong task	0.2296 ± 0.3043	0.5803 ± 0.1090	0.2562 ± 0.2932	0.2045 ± 0.4122	0.5908 ± 0.1809	0.4719 ± 0.1373
Random marginal	-0.0436 ± 0.2413	0.4849 ± 0.0835	0.1322 ± 0.0492	-0.0086 ± 0.3205	0.4954 ± 0.1290	0.3510 ± 0.0500
Constant	N/A	0.5000 ± 0.0000	0.1348 ± 0.0493	N/A	0.5000 ± 0.0000	0.3484 ± 0.0476

C Behavior-Cloning Protocol

Candidate pools. BC data are separate from the reward-prediction benchmark; source groups keep related trajectory versions together during splitting. MetaWorld MT10 includes button-press-topdown, door-open, drawer-close, drawer-open, peg-insert-side, pick-place, push, reach, window-close, and window-open (all v3). Each task has 100 source groups, each with successful, high-, medium-, and low-progress versions from related initial conditions. ManiSkill groups contain a clean official replay and three noisy versions (noise scales 0.03, 0.08, 0.16; AR(1) coefficient 0.85). Perturbed actions are executed in the environment and serve as BC targets. Per seed, 20 groups form the context reservoir; the remaining 80 supply four versions each, totaling 320 candidates. Ten reservoir groups supply ten context points each, preventing source-group overlap with candidates.

Selection. Selectors share the candidate pool within each seed and select $K \in \{20, 40, 60\}$ trajectories. GRIP combines the 20th percentile of predicted rewards over each trajectory's final 10% with visual diversity. Random samples uniformly; diversity-only selection omits rewards; GT uses environment rewards for the same tail score. All uses all 320 candidates.

Policy training. BC policies take states and task text as input and predict executed actions; predicted rewards are not inputs. Each trajectory has equal total MSE weight regardless of length. MetaWorld uses an MLP with two 256-unit hidden layers and 15,000 updates, evaluated on 50 new variants per task (500 rollouts). ManiSkill uses three 1,024-unit hidden layers and 10,000 updates, evaluated on 100 new episodes per task (800 rollouts). Both use batch size 1,024 and AdamW (learning rate 10^{-3}, weight decay 10^{-4}).

Selectors share source-group splits, policy initialization, update budgets, and evaluation seeds. We report means and sample standard deviations over three paired seeds, including variation from both selection and BC training. All also uses a longer schedule matched to $K = 40$ data exposure: 120,000 MetaWorld updates and 80,000 ManiSkill updates.

The longer All schedule yields lower mean success on both benchmarks (Table 11), so its gap to GRIP is not explained by too few updates alone. Comparisons with All change both data quantity

Table 10: Training-free GRIP with a growing shared MetaWorld context. Each task contributes 100 labeled points, and every query task is represented. Total context therefore grows with task count rather than remaining fixed. Values are task-level macro means ± sample standard deviations; GRIP uses the same parameters throughout.

Tasks	Total context points	Spearman ↑	C-index ↑	NRMSE ↓
1	100	0.6795 ± 0.1628	0.7660 ± 0.0747	0.0910 ± 0.0374
2	200	0.6710 ± 0.1269	0.7615 ± 0.0756	0.0908 ± 0.0297
5	500	0.6750 ± 0.0768	0.7642 ± 0.0746	0.0906 ± 0.0181
10	1,000	0.6783 ± 0.0556	0.7660 ± 0.0746	0.0918 ± 0.0126
20	2,000	0.6791 ± 0.0212	0.7665 ± 0.0784	0.0943 ± 0.0071
50	5,000	0.6809 ± 0.0061	0.7675 ± 0.0767	0.0909 ± 0.0029

Table 11: BC success (%) for trajectory selection and two All training schedules. Values are means ± sample standard deviations over three paired seeds. All uses 320 trajectories per task. Compute-matched training uses 15,000/10,000 updates on MetaWorld/ManiSkill; matching $K = 40$ data exposure uses 120,000/80,000 updates. The two All rows do not correspond to a particular K.

	MetaWorld MT10			ManiSkill		
Selection method	K=20	K=40	K=60	K=20	K=40	K=60
Random	36.5 ± 9.2	48.6 ± 7.0	39.5 ± 11.6	31.8 ± 2.3	37.2 ± 1.1	38.9 ± 1.9
Visual diversity	36.8 ± 2.6	43.9 ± 10.6	34.9 ± 15.7	33.6 ± 6.5	39.6 ± 1.3	41.2 ± 3.8
GT reward	90.7 ± 1.7	94.9 ± 1.9	97.8 ± 1.2	27.8 ± 1.2	36.3 ± 2.1	41.5 ± 0.4
GRIP	**94.6 ± 3.6**	**95.7 ± 1.5**	**96.2 ± 0.2**	**35.7 ± 3.4**	**40.4 ± 3.1**	**44.1 ± 3.5**
All (compute-matched)			53.4 ± 14.2			41.6 ± 3.6
All ($K = 40$ exposure-matched)			44.9 ± 8.5			38.2 ± 1.8

and quality; comparisons with GT assess selection outcomes, not whether predicted rewards are intrinsically better than environment rewards.