TUCO: CURATING SIMULATION DEMONSTRATIONS FOR SIM-TO-REAL ROBOT POLICY CO-TRAINING

Anonymous authors
Paper under double-blind review

ABSTRACT

Simulation demonstrations can supplement scarce real-world data for robot policy co-training. However, the value of using data curation to actively select these demonstrations for sim-to-real co-training remains underexplored. Existing curation methods also lack a unified criterion for measuring trajectory-level utility and set-level coverage from closed-loop target behavior. To address these gaps, we present the first systematic study of data curation for sim-to-real robot policy co-training and propose **T**rajectory-level **U**tility and set-level **C**overage **O**ptimization (**TUCO**). TUCO uses influence functions to trace how each source demonstration affects target-domain scoring rollouts. Our key insight is that these effects can be decomposed into an overall contribution to target return and variation across rollouts, providing a common closed-loop basis for measuring trajectory utility and set coverage. We further propose a performance-aligned subset optimizer that combines these measures in a unified curation objective to reduce redundancy and select complementary demonstrations. Extensive experiments on RoboMimic and OmniReset establish the value of active simulation data curation for sim-to-real policy co-training and show that TUCO achieves state-of-the-art performance across single-simulator, sim-to-sim, and sim-to-real settings.

1 INTRODUCTION

Imitation learning is essential for robot manipulation, but the amount of available training data is limited by the cost of real-world teleoperation (Mandlekar et al., 2022; Belkhale et al., 2023; Lin et al., 2025a; Saxena et al., 2025). Automated data generation in simulation offers a scalable alternative (Mandlekar et al., 2023; Nasiriany et al., 2024), motivating sim-to-real co-training that combines abundant simulation demonstrations from a source domain with limited real demonstrations from the target deployment domain (Maddukuri et al., 2025; Cheng et al., 2025). However, domain shifts make source demonstrations differ in their utility to the target policy, while large source pools may include redundant or simulation-specific behaviors. Directly using the full pool can therefore increase training costs and introduce negative transfer (Dass et al., 2026; Wei et al., 2026). This raises a central but unaddressed question for sim-to-real co-training: *which simulation demonstrations should be selected to improve target-domain policy performance?*

Robot data curation aims to improve policy performance by constructing training subsets that exclude unhelpful or redundant demonstrations, with existing methods broadly falling into three groups. (1) **Quality-based approaches** filter noisy demonstrations using data statistics and execution cues (Belkhale et al., 2023; Hejna et al., 2025b; Zhang et al., 2026; Sojib et al., 2026). They offer efficient offline selection but provide no direct link to closed-loop performance (Bedi, 2026). (2) **Utility-based approaches** estimate downstream value from target similarity, validation loss, or policy behavior (Chen et al., 2025; Du et al., 2023; Memmel et al., 2025; Dass et al., 2026; Lee et al., 2026). **Influence Function**-based methods are highly promising since they can estimate how reweighting a training example changes the learned model and a downstream metric (Koh & Liang, 2017). CUPID (Agia et al., 2025) applies influence functions to closed-loop evaluation by aggregating each demonstration's effects across policy rollouts with their returns. Although the resulting scalar utility connects data selection to target performance, independent ranking does not account for relationships within the selected subset and can select redundant demonstrations while overlooking complementary ones. (3) **Diversity-based approaches** explicitly model these relationships to reduce redundancy and improve coverage (Zhang et al., 2026; Sirigiri et al., 2026; Jia et al., 2024),

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.

but their criteria typically measure differences in representation space without grounding them in closed-loop target behavior. Under domain shift, such differences may reflect simulator-specific variation that offers little support for target-policy execution. These limitations yield another important question: *How can trajectory-level utility and set-level coverage be jointly optimized in closed-loop performance?*

Our key observation is that the influence of a demonstration across closed-loop target-policy rollouts can be explicitly decomposed into two signals: an overall contribution to target return and the remaining variation across rollouts, providing a common closed-loop basis for measuring trajectory-level utility and set-level coverage. The overall contribution indicates how much a demonstration supports target performance, while relationships among the remaining components reveal redundancy and complementarity across demonstrations. However, since variation across rollouts alone does not imply a positive contribution to target performance, subset selection should retain demonstrations whose effects align with target success and favor candidates that add new closed-loop support beyond the current subset.

Beyond these methodological limitations, existing data curation research has a limited evaluation scope. Most studies focus on single-domain settings or transfer between simulators, or among real-world datasets (Du et al., 2023; Lin et al., 2025b; Memmel et al., 2025; Hejna et al., 2025a; Dass et al., 2026; Yang et al., 2026a), leaving sim-to-real policy co-training underexplored. Meanwhile, sim-to-real co-training studies primarily examine mixing ratios and training strategies (Maddukuri et al., 2025; Cheng et al., 2025; Wei et al., 2026), with limited attention to which simulation demonstrations best support real-world policy learning. We aim to extend data curation from single-simulator and sim-to-sim settings to sim-to-real co-training through a unified protocol. Within this broader evaluation, we explore whether actively selecting simulation demonstrations improves real-world policy performance when the numbers of source and target demonstrations are held fixed.

To our knowledge, we conduct the first systematic study of simulation data curation for sim-to-real robot policy co-training and introduce **T**rajectory-level **U**tility and set-level **C**overage **O**ptimization (**TUCO**), a novel set-level curation framework grounded in target-policy closed-loop behavior. First, we present a closed-loop influence decomposition that separates each demonstration's overall contribution to target return from its variation across target-domain scoring rollouts, providing a common basis for measuring trajectory-level utility and set-level coverage. Second, we introduce a performance-aligned subset optimizer that combines both measures in a unified curation objective. It favors demonstrations that support target success and complement the current subset, reducing redundancy during selection. We establish a unified evaluation protocol on RoboMimic (Mandlekar et al., 2022) and OmniReset (Yin et al., 2026) spanning single-simulator, sim-to-sim, and sim-to-real settings and show that TUCO achieves state-of-the-art curation performance. More importantly, we show that active simulation data curation improves sim-to-real policy co-training: TUCO improves over full-pool co-training by 65.0, 35.0, and 80.0 percentage points (pp) across the three real-world tasks and outperforms random selection with the same numbers of source and target demonstrations.

2 Related Work

Data Curation for Robot Learning. Existing approaches assess demonstrations through quality, utility, or diversity. Quality-based methods use statistical dependence (Hejna et al., 2025b) and execution cues (Zhang et al., 2026; Sojib et al., 2026) to estimate demonstration quality. Utility-based methods condition selection on target-data similarity (Du et al., 2023; Memmel et al., 2025) or estimated training contributions (Chen et al., 2025; Dass et al., 2026), with influence functions relating these contributions to imitation loss (Lee et al., 2026) and closed-loop return (Agia et al., 2025; Xu et al., 2026). Diversity-based methods construct subsets through deduplication and trajectory diversity (Zhang et al., 2026; Sirigiri et al., 2026) or distribution matching (Feng et al., 2025).

Cross-Domain Robot Co-Training. Simulation data generation (Mandlekar et al., 2023; Nasiriany et al., 2024; Zhao et al., 2026) supports co-training with limited real demonstrations. Composition-oriented studies examine simulation scale and mixing ratios (Wei et al., 2025; Maddukuri et al., 2025), while alignment methods improve source-target compatibility through observation-action alignment (Cheng et al., 2025) and visual invariance (Yang et al., 2025). Complementary approaches regulate source contributions through learned mixture weights (Hejna et al., 2025a; Sima et al., 2026) and training schedules (Wei et al., 2026).

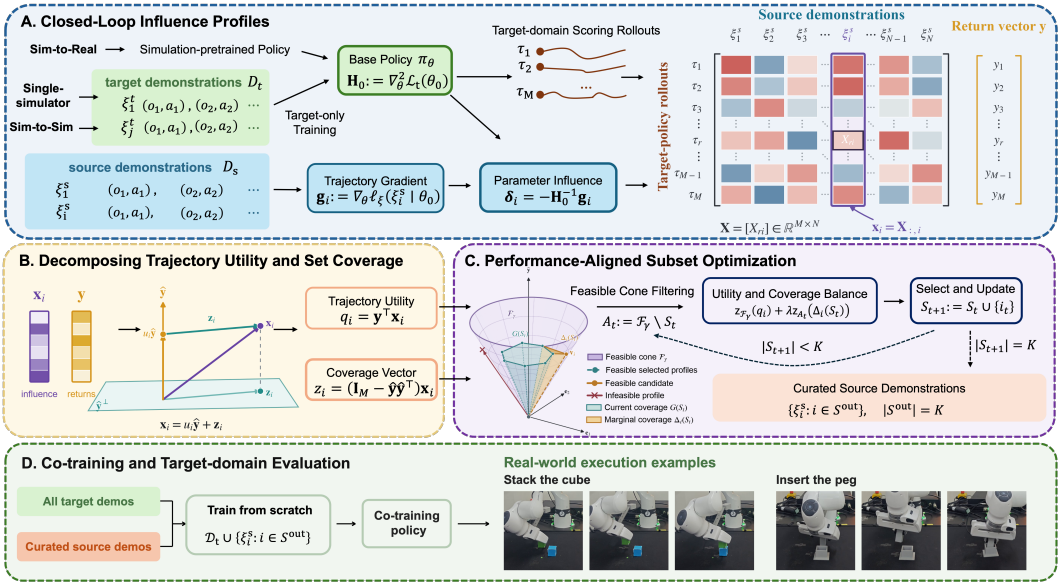

Figure 1: **Overview of TUCO.** (**A**) Influence functions estimate source demonstration effects on target-domain scoring rollouts at a fixed base policy, yielding the influence matrix $\mathbf{X}$ and return vector $\mathbf{y}$. (**B**) Each influence profile is decomposed for trajectory utility and set coverage. (**C**) TUCO filters candidates by performance alignment and iteratively selects trajectories using utility and marginal coverage. (**D**) The curated source subset is combined with target demonstrations for co-training and target-domain evaluation.

3 Preliminaries: Data Attribution via Influence Functions

Effective data curation depends on understanding how each training sample affects the output. **Data Attribution** quantifies this dependence by estimating how changes to training data affect a model's performance. Evaluating these effects through repeated retraining is expensive, so **Influence Functions** estimate them locally by perturbing the weight of a training example (Koh & Liang, 2017). Consider a training set $\mathcal{D} := \{z_i\}_{i=1}^n$ with per-example loss $\ell(z;\theta)$ and parameters $\theta \in \mathbb{R}^{d_\theta}$. The learned parameters $\theta_0 := \theta(\mathcal{D})$ minimize $\mathcal{L}(\theta;\mathcal{D}) := n^{-1}\sum_{i=1}^n \ell(z_i;\theta)$. Increasing the weight of an example z by ϵ gives the perturbed objective $\mathcal{L}(\theta;\mathcal{D}) + \epsilon\ell(z;\theta)$, whose local minimizer near θ_0 is denoted by $\theta_{\epsilon,z}$. Let $f(\hat{z};\theta)$ be a differentiable metric evaluated on an example $\hat{z}$ and $\mathbf{H}_0 := \nabla^2_\theta \mathcal{L}(\theta_0;\mathcal{D}) \in \mathbb{R}^{d_\theta \times d_\theta}$ the training-loss Hessian. With twice continuously differentiable training losses and a nonsingular $\mathbf{H}_0$, the influence of z on this metric is $\Psi_{\text{inf}}(\hat{z},z) := \partial_\epsilon f(\hat{z};\theta_{\epsilon,z})|_{\epsilon=0} = -\nabla_\theta f(\hat{z};\theta_0)^\top \mathbf{H}_0^{-1} \nabla_\theta \ell(z;\theta_0)$. This measures how upweighting a training example changes the evaluation metric through its effect on the learned parameters.

4 Methodology

In this section, we present TUCO for source demonstration selection in target-domain policy co-training. First, we formulate the source-to-target co-training problem. Second, we decompose closed-loop influence profiles to measure trajectory utility and set coverage. Third, we combine both measures in a performance-aligned subset optimizer. Fig. 1 provides an overview.

4.1 Problem Formulation

Imitation Learning. We consider a source environment $\mathcal{E}_s$ and a target environment $\mathcal{E}_t$ with compatible observation and action interfaces but potentially different observations or dynamics. A demonstration $\xi = ((o_h, a_h))_{h=0}^{T_\xi - 1}$ is an observation-action trajectory, where $o_h \in \mathcal{O} \subseteq \mathbb{R}^{d_o}$ and $a_h \in \mathcal{A} \subseteq \mathbb{R}^{d_a}$. Given a demonstration set $\mathcal{D}$, behavior cloning (Argall et al., 2009; Osa et al., 2018) trains a policy π_θ with $\theta \in \mathbb{R}^{d_\theta}$ by minimizing $\theta(\mathcal{D}) \in$

$\arg\min_{\theta'} \left\{\mathcal{L}_{\mathrm{IL}}(\theta', \mathcal{D}) := \mathbb{E}_{(o,a)\sim\nu_{\mathcal{D}}}\left[\ell(o, a \mid \theta')\right]\right\}$. $\nu_{\mathcal{D}}$ is the observation-action sampling distribution. We evaluate the policy in $\mathcal{E}_{\mathrm{t}}$ through its expected return $J_{\mathrm{t}}(\theta) := \mathbb{E}_{\tau\sim p_{\mathrm{t}}(\tau|\pi_\theta)}[R_{\mathrm{t}}(\tau)]$, where p_{t} denotes the target rollout distribution and R_{t} the task return. For binary tasks, we assign return $+1$ to success and -1 to failure.

Source-to-Target Data Curation. Let $\mathcal{D}_{\mathrm{t}} := \{\xi_j^{\mathrm{t}}\}_{j=1}^{n_{\mathrm{t}}}$ denote the fixed target demonstrations and $\mathcal{D}_{\mathrm{s}} := \{\xi_i^{\mathrm{s}}\}_{i=1}^{N}$ the source candidate pool. Given a budget $K \leq N$, we select an index set $S \subseteq [N] := \{1, \ldots, N\}$ of size K and combine the corresponding trajectories $\mathcal{D}_{\mathrm{s}}(S) := \{\xi_i^{\mathrm{s}} : i \in S\}$ with all target demonstrations for policy training. In sim-to-real comparisons, we hold the source-to-target sampling ratio fixed across curation methods. The curation objective is $S^\star \in \arg\max_{S\subseteq[N],|S|=K} J_{\mathrm{t}}\left(\theta(\mathcal{D}_{\mathrm{t}} \cup \mathcal{D}_{\mathrm{s}}(S))\right)$.

This formulation covers three source-to-target settings: (1) **Single-Simulator**, where all demonstrations come from the same simulated environment. We evaluate both selection, which augments a fixed target set with selected candidates, and filtering, which retains a subset of an existing training set; (2) **Sim-to-Sim**, where source and target demonstrations come from different simulators; and (3) **Sim-to-Real**, where source demonstrations come from simulation and target demonstrations from the physical environment.

4.2 Closed-Loop Influence Profiles

We first obtain a base policy π_{θ_0} and use its closed-loop executions in the target environment $\mathcal{E}_{\mathrm{t}}$ as feedback for curation. In single-simulator and sim-to-sim settings, we train this policy on target demonstrations and collect its rollouts, using the original unfiltered training set for single-simulator filtering. In sim-to-real, we use a simulation-pretrained policy and reuse its successful rollouts recorded during real-robot deployment. We keep the base policy fixed and score all source candidates against the same set of target-domain scoring rollouts $\mathcal{R}_{\mathrm{t}} := \{\tau_r\}_{r=1}^{M}$. We use influence functions to estimate how each source demonstration affects these target executions. For a source trajectory of length T_i, let $\ell_\xi(\xi_i^{\mathrm{s}} \mid \theta) := \sum_{h=0}^{T_i-1} \ell(o_{i,h}^{\mathrm{s}}, a_{i,h}^{\mathrm{s}} \mid \theta)$ denote its training loss. We first derive the profile for a target-trained base policy $\theta_0 := \theta(\mathcal{D}_{\mathrm{t}})$. Increasing the source trajectory's weight in the target training objective $\mathcal{L}_{\mathrm{t}}(\theta) := \mathcal{L}_{\mathrm{IL}}(\theta, \mathcal{D}_{\mathrm{t}})$ gives the local minimizer $\theta_{\epsilon,i}$ of $\mathcal{L}_{\mathrm{t}}(\theta) + \epsilon\ell_\xi(\xi_i^{\mathrm{s}} \mid \theta)$ near θ_0. For a target rollout τ_r, the influence expression gives $X_{ri} := \partial_\epsilon \log p_{\mathrm{t}}(\tau_r \mid \pi_{\theta_{\epsilon,i}})\big|_{\epsilon=0} = -\mathbf{h}_r^\top \mathbf{H}_0^{-1}\mathbf{g}_i$, where $\mathbf{g}_i := \nabla_\theta \ell_\xi(\xi_i^{\mathrm{s}} \mid \theta_0) \in \mathbb{R}^{d_\theta}$ is the source trajectory's training gradient, $\mathbf{H}_0 := \nabla_\theta^2 \mathcal{L}_{\mathrm{t}}(\theta_0) \in \mathbb{R}^{d_\theta \times d_\theta}$ is the target training-loss Hessian and $\mathbf{h}_r := \nabla_\theta \log p_{\mathrm{t}}(\tau_r \mid \pi_\theta)|_{\theta=\theta_0} \in \mathbb{R}^{d_\theta}$ is the rollout log-likelihood gradient. A positive X_{ri} means that locally upweighting demonstration i makes rollout r more likely. Collecting these effects over $\mathcal{R}_{\mathrm{t}}$ gives the matrix $\mathbf{X} := [X_{ri}] \in \mathbb{R}^{M\times N}$. Its column $\mathbf{x}_i := \mathbf{X}_{:,i} \in \mathbb{R}^M$ forms the rollout-conditioned influence profile of demonstration i. The profile records which executions become more or less likely, and their returns determine resulting performance contributions. With $y_r := R_{\mathrm{t}}(\tau_r)$ and $\mathbf{y} := [y_1, \ldots, y_M]^\top \in \mathbb{R}^M$, we obtain $q_i := \mathbf{y}^\top \mathbf{x}_i = \sum_{r=1}^{M} y_r X_{ri}$. For independent rollouts sampled from $p_{\mathrm{t}}(\cdot \mid \pi_{\theta_0})$, q_i/M estimates $\partial_\epsilon J_{\mathrm{t}}(\theta_{\epsilon,i})|_{\epsilon=0}$ when profile entries are exact and the conditions in Appendix B.1 hold. In practice, we construct $\mathbf{X}$ using the projected loss-gradient estimator in Appendix B.2. For sim-to-real curation, we evaluate these gradients at the frozen simulation-pretrained policy and set $y_r = 1$ for each recorded successful execution, so q_i measures estimated support for the observed successful target behaviors. Our key insight is that each influence profile can be decomposed into a component along the target-return direction and an orthogonal component. The former preserves trajectory-level utility, while the latter captures differences in rollout influence among demonstrations. This decomposition provides a common closed-loop basis for measuring individual utility and complementary coverage within a selected set.

4.3 Decomposing Trajectory Utility and Set Coverage

The utility q_i supports trajectory ranking through CUPID's linear subset objective $F_{\mathrm{CUPID}}(S) := \sum_{i\in S} q_i$, which is maximized by Top-K selection. However, aggregation into q_i leaves differences orthogonal to the return vector unrepresented, so distinct rollout influence profiles can receive the same utility. Each trajectory also contributes a fixed q_i regardless of the selected set, preventing these differences from influencing whether trajectories are selected together. We therefore introduce a closed-loop influence decomposition that preserves trajectory utility while exposing the remaining

differences across target-domain scoring rollouts. Let $\widehat{\mathbf{y}} := \mathbf{y}/\|\mathbf{y}\|_2 \in \mathbb{R}^M$ and let $\mathbf{I}_M \in \mathbb{R}^{M\times M}$ denote the identity matrix. We decompose

$$\mathbf{x}_i = \underbrace{u_i\widehat{\mathbf{y}}}_{\text{utility component}} + \underbrace{\mathbf{z}_i}_{\text{coverage representation}}, \qquad u_i := \widehat{\mathbf{y}}^\top \mathbf{x}_i, \qquad \mathbf{z}_i := \left(\mathbf{I}_M - \widehat{\mathbf{y}}\widehat{\mathbf{y}}^\top\right)\mathbf{x}_i. \tag{1}$$

The coordinate u_i preserves trajectory-level utility through $q_i = \|\mathbf{y}\|_2 u_i$. The residual $\mathbf{z}_i \in \mathbb{R}^M$ captures how trajectories with similar utility affect scoring rollouts differently, providing a basis for measuring their redundancy and complementarity. We use these residual directions to measure the additional coverage a candidate provides relative to the selected set.

4.4 Performance-Aligned Subset Optimization

The residuals reveal differences across target-domain scoring rollouts, but covering these differences is useful only when the selected demonstrations also support target performance. We therefore develop a performance-aligned subset optimizer to combine this coverage with trajectory utility. We first restrict coverage optimization to candidates with positive estimated trajectory-utility and sufficient alignment with $\mathbf{y}$, defining the feasible set

$$\mathcal{F}_\gamma := \left\{ i \in [N] : q_i > 0,\ \|\mathbf{x}_i\|_2 > 0,\ \frac{\mathbf{x}_i^\top \mathbf{y}}{\|\mathbf{x}_i\|_2\|\mathbf{y}\|_2} \geq \gamma \right\}, \tag{2}$$

where $\gamma \in [0, 1)$ controls the width of the cone around the target-return direction. Through this constraint, we require the return-aligned component to account for at least a fraction γ of the influence-profile norm. Within this feasible pool, we aim to improve set-level coverage by selecting demonstrations that contribute influence directions poorly represented in the current subset. The intuition is to view coverage as the volume spanned by the selected residual vectors, so a demonstration contributes through the additional volume it introduces. To focus on directional coverage, we normalize residuals to $\mathbf{v}_i := \mathbf{z}_i/\|\mathbf{z}_i\|_2 \in \mathbb{R}^M$. We also give higher-utility trajectories greater weight through $w_i := q_i/\kappa_{\mathcal{F}}$, where $\kappa_{\mathcal{F}} := |\mathcal{F}_\gamma|^{-1}\sum_{j\in\mathcal{F}_\gamma} q_j$. We formalize this intuition with a quality-weighted, regularized log-determinant criterion and its marginal gain:

$$G(S) := \log\det\left(\mathbf{I}_M + \rho\sum_{i\in S} w_i \mathbf{v}_i\mathbf{v}_i^\top\right), \qquad \Delta_i(S) := G(S\cup\{i\}) - G(S), \tag{3}$$

where $S \subseteq \mathcal{F}_\gamma$ and $\rho > 0$ controls the coverage scale. For comparable utility weights, this criterion gives smaller gains to directions already well represented in S. The identity term keeps the measure well defined for subsets of any size. We provide the precise volume relationship and efficient computation in Appendix B.3. Because a candidate's additional coverage depends on the demonstrations already selected, we update its marginal gain as the subset grows. We combine this gain with trajectory utility to favor demonstrations that are both useful and complementary. To put the two terms on comparable scales, we use population z-scores z_A over the fixed feasible pool for utility and the remaining candidates for coverage. We assign zero when the corresponding variance vanishes. Starting from $S_0 = \varnothing$, we select from $A_t := \mathcal{F}_\gamma \setminus S_t$ according to

$$i_t \in \arg\max_{i\in A_t}\left[\mathrm{z}_{\mathcal{F}_\gamma}(q_i) + \lambda \mathrm{z}_{A_t}(\Delta_i(S_t))\right], \qquad S_{t+1} := S_t \cup \{i_t\}, \tag{4}$$

where $\lambda \geq 0$ balances utility and coverage. We keep the influence profiles fixed throughout selection and recompute marginal gains after each addition until the budget or feasible pool is exhausted. When $|\mathcal{F}_\gamma| < K$, we retain all feasible candidates and fill the remaining slots with the highest-utility unselected trajectories. For co-training, we retrain the policy on all fixed target demonstrations with the selected source demonstrations, $\mathcal{D}_{\mathrm{t}} \cup \mathcal{D}_{\mathrm{s}}(S^{\mathrm{out}})$, see Algorithm 1 for summary.

5 Experiments

We conduct a series of experiments to compare TUCO with state-of-the-art robot data curation baselines across Single-Simulator, Sim-to-Sim, and Sim-to-Real co-training. Following CUPID (Agia et al., 2025), the Single-Simulator study uses the `Lift`, `Square`, and `Transport` tasks from RoboMimic's multi-human (MH) task suite (Mandlekar et al., 2022). The cross-domain studies use the `Peg`, `StackCube`, and `CupCake` tasks from OmniReset (Yin et al., 2026). We aim to answer: (**Q1**) Can TUCO enable reliable curation as the domain gap changes? (**Q2**) Can active data curation improve sim-to-real co-training performance?

Algorithm 1 TUCO: Trajectory-Level Utility and Set-Level Coverage Optimization

Require: Influence matrix $\mathbf{X} \in \mathbb{R}^{M \times N}$, returns $\mathbf{y} \in \mathbb{R}^{M}$, budget K, parameters γ, λ, ρ
Ensure: Selected source indices $S^{\text{out}} \subseteq [N]$
$\widehat{\mathbf{y}} \leftarrow \mathbf{y}/\|\mathbf{y}\|_2$
$q_i \leftarrow \mathbf{y}^\top \mathbf{x}_i$, $\mathbf{z}_i \leftarrow \mathbf{x}_i - (\widehat{\mathbf{y}}^\top \mathbf{x}_i)\widehat{\mathbf{y}}$ for $i \in [N]$
$\mathcal{F}_\gamma \leftarrow \{i \in [N] : q_i > 0,\ \|\mathbf{x}_i\|_2 > 0,\ \mathbf{x}_i^\top \widehat{\mathbf{y}}/\|\mathbf{x}_i\|_2 \geq \gamma\}$
Normalize residuals to $\mathbf{v}_i$ and utilities to w_i over $\mathcal{F}_\gamma$ when nonempty
$S \leftarrow \varnothing$, $\mathbf{M} \leftarrow \mathbf{I}_M \in \mathbb{R}^{M \times M}$
while $|S| < \min(K, |\mathcal{F}_\gamma|)$ **do**
  $A \leftarrow \mathcal{F}_\gamma \setminus S$
  $\Delta_i \leftarrow \log(1 + \rho w_i \mathbf{v}_i^\top \mathbf{M}^{-1} \mathbf{v}_i)$ for $i \in A$
  $i^\star \leftarrow \arg\max_{i \in A} [\mathbf{z}_{\mathcal{F}_\gamma}(q_i) + \lambda \mathbf{z}_A(\Delta_i)]$
  $\mathbf{M} \leftarrow \mathbf{M} + \rho w_{i^\star} \mathbf{v}_{i^\star} \mathbf{v}_{i^\star}^\top$, $S \leftarrow S \cup \{i^\star\}$
end while
if $|S| < K$ **then**
  Add the $K - |S|$ highest-utility indices from $[N] \setminus S$ to S
end if
return $S^{\text{out}} \leftarrow S$

Experimental Settings. We follow the source-to-target curation problem defined in Section 4. In the **single-simulator setting**, source demonstrations and target-policy evaluation share the same RoboSuite environment. Following CUPID (Agia et al., 2025), we separately partition each 300-demonstration RoboMimic dataset for filtering and selection. Filtering removes demonstrations from an initial set of 192, with 12 reserved for validation. Selection augments 48 fixed target demonstrations with a subset of 240 source candidates, using the remaining 12 for validation. In the **sim-to-sim setting**, IsaacSim is the source domain and MuJoCo is the target domain. `Peg` uses 3,000 source and 500 target demonstrations, while `StackCube` and `CupCake` each use 5,000 source and 1,000 target demonstrations. In both simulated settings, each base policy provides 100 target-domain scoring rollouts. In the **sim-to-real setting**, we reuse 10 recorded successful real-world rollouts per task from a frozen simulation-pretrained base policy. These serve as both the target demonstrations for co-training and the scoring rollouts for curation. All curation methods select 600 demonstrations from the same pool of 1,200 simulation demonstrations. For TUCO, we fix the cone threshold at $\gamma = 0.01$, the coverage scale $\rho = 0.1$ and use $\lambda = 0.2$ throughout. We refer to the detailed setup to Appendix C.

Baselines and Evaluation Protocol. For single-simulator and sim-to-sim settings, we compare with representative methods from the three main families of robot data curation. **Quality-based** baselines include DemInf (Hejna et al., 2025b) and PSD (Sojib et al., 2026). **Utility-based** baselines include Demo-SCORE (Chen et al., 2025), Success Similarity and CUPID (Agia et al., 2025), QoQ (Lee et al., 2026), and DataMIL (Dass et al., 2026). **Diversity-based** baselines include FAKTUAL (Sirigiri et al., 2026), SIEVE (Wu et al., 2026), and TAROT (Feng et al., 2025). For sim-to-real settings, we compare with Success Similarity, CUPID (Agia et al., 2025), DataMIL (Dass et al., 2026) and FAKTUAL (Sirigiri et al., 2026). [1]We also report random selection together with target-only training and co-training on the full source pool. Each selected subset is used to train a policy from scratch, and performance is measured by closed-loop success rate in the target domain.

5.1 Main Results: From Single-Simulator Curation to Sim-to-Real Co-Training

Single-Simulator. We first evaluate TUCO under the standard single-simulator protocols, which isolate data curation from domain shift. Figure 2 shows that TUCO achieves the highest or tied-highest mean success rate in 18 of 25 task–budget combinations. On `Square`, TUCO improves over CUPID by 14.2 pp after filtering 50% of the demonstrations and by 15.8 pp when selecting 10% of the candidates. Notably, TUCO matches full-data training on `Lift` with only 25% of the original training set and surpasses it on `Square` with 50%. These results establish the value of curation in a controlled setting and motivate its evaluation in the cross-domain settings that follow.

[1]Given real-robot costs, we prioritize baselines closely aligned with target-conditioned subset selection. Demo-SCORE requires unavailable failure feedback. Broader comparisons are conducted in simulation.

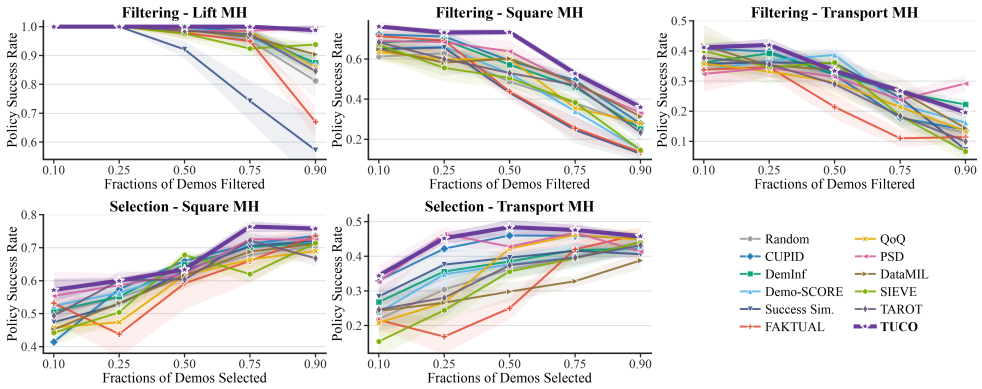

Figure 2: Single-simulator data curation on RoboMimic under demonstration filtering and selection. Results are averaged over 3 random seeds. Success rates are computed over 50 evaluation rollouts from the last 10 checkpoints (500 rollouts total).

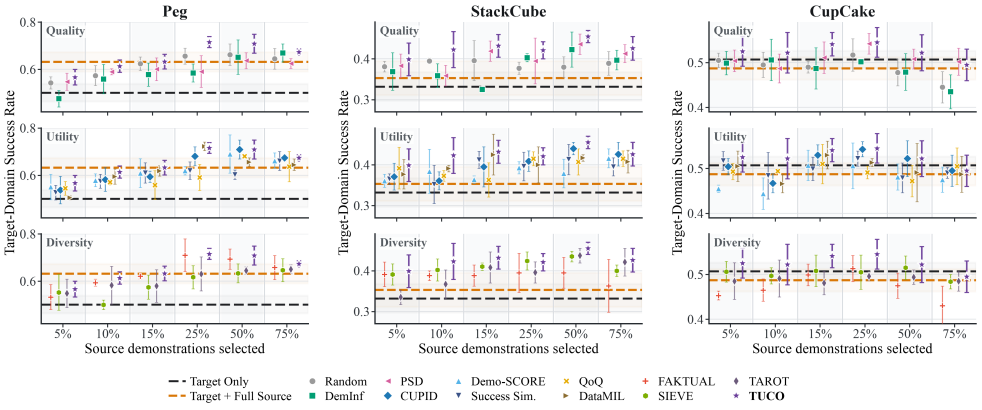

Figure 3: Sim-to-Sim co-training with curated IsaacSim demonstrations and fixed MuJoCo target data across different source-data budgets. Results are averaged over 3 random seeds. Success rates are computed over 50 evaluation rollouts from the last 5 checkpoints (250 rollouts total).

Sim-to-Sim Co-Training. We first examine cross-domain curation in Sim-to-Sim as a controlled setting with observation and dynamics mismatch. Figure 3 shows that TUCO achieves the highest mean success rate in 15 of the 18 task–budget combinations and outperforms random selection in all 18. Its best curated subsets exceed full-pool co-training by 8.4 pp on `Peg`, 10.2 pp on `StackCube`, and 5.8 pp on `CupCake`. Full-pool co-training improves `Peg`, offers only a modest gain on `StackCube`, and causes negative transfer on `CupCake`. In contrast, each task reaches its best performance with a curated subset. This shows that successful source demonstrations can differ substantially in their target-domain value under simulator shift. The comparison with random selection further shows that the gain comes from identifying suitable trajectories rather than merely reducing the amount of source data. Quality-based methods remain inconsistent because source-domain quality does not directly indicate transfer value. Utility-based methods better reflect the target task but can retain redundant trajectories through independent ranking. Diversity-based methods perform well at some budgets, although source-space variation does not always support target-domain behavior. TUCO is more consistent by grounding both utility and coverage in target-domain scoring rollouts.

Sim-to-Real Co-Training. Finally, we examine whether active curation improves co-training on physical manipulation tasks. Table 1 shows that TUCO achieves the highest success rate on all three tasks. Although FAKTUAL matches TUCO on `StackCube` and `CupCake`, its success rate is 72.5 pp lower on `Peg`, highlighting the consistency of TUCO across tasks. More importantly, TUCO

Table 1: Success rates (%) across three real-world tasks. Each method is evaluated over 40 trials per task (120 real-robot trials in total). Avg. denotes the macro-average across tasks.

Method / training data	`Peg`	`StackCube`	`CupCake`	**Avg.**
Training-data baselines				
10 real demos + all sim data	17.5	40.0	20.0	25.8
10 real demos	15.0	35.0	0.0	16.7
Compared methods				
Random	7.5	57.5	12.5	25.8
CUPID	75.0	65.0	80.0	73.3
DataMIL	22.5	65.0	**100.0**	62.5
FAKTUAL	10.0	**75.0**	**100.0**	61.7
Success Similarity	5.0	50.0	60.0	38.3
TUCO (ours)	$\mathbf{82.5}^{\dagger\ddagger}$	$\mathbf{75.0}^{\dagger}$	$\mathbf{100.0}^{\dagger\ddagger}$	**85.8**

For TUCO, † and ‡ indicate significant improvements over full-pool co-training and random selection, respectively (two-sided Fisher's exact tests with Holm correction across six comparisons, adjusted $p < 0.05$).

improves over full-pool co-training by 65.0 pp, 35.0 pp, and 80.0 pp on `Peg`, `StackCube`, and `CupCake`, respectively. With the same numbers of source and target demonstrations, TUCO also outperforms random selection by 75.0 pp, 17.5 pp, and 87.5 pp supporting source-data composition as a consequential design choice in sim-to-real co-training. These gains are obtained using only 10 recorded successful real-world rollouts per task from a simulation-pretrained base policy. While the simulated experiments collect scoring rollouts from a target-trained policy, the real-world experiments reuse these existing executions as both scoring feedback and target demonstrations. Even with identical success labels, their influence profiles retain differences in how source demonstrations support individual executions, allowing utility and coverage to guide selection. The results therefore show that limited successful real-world experience can provide useful feedback for simulation data curation without additional scoring rollouts.

5.2 Ablation Results

We further ablate TUCO to examine the contributions of trajectory utility, set coverage, and the cone constraint. **Utility** uses closed-loop utility ranking and corresponds to CUPID. **Coverage + Cone** removes the additive utility term and selects by quality-weighted marginal coverage. Once the cone is exhausted,it continues this selection outside the cone, retaining signed utility weights. **Utility + Coverage** sets $\gamma = 0$ while retaining both terms in the selection objective. Table 2 compares these variants with TUCO, where U, C, and G denote utility, coverage, and the cone constraint. TUCO achieves the highest in 15 of the 18 settings. Compared with U, it improves mean performance by 2.1 pp on average. The gain increases to 3.0 pp for budgets up to 25%, suggesting that coverage is particularly useful when selection is restrictive. The coverage-driven variant yields lower mean performance in 16 settings. Setting $\gamma = 0$ lowers it in 12 settings. These results indicate that coverage benefits from direct utility guidance and a performance-aligned candidate pool. At the 75% budget, TUCO and $U + C$ exhaust their feasible pools and complete selection by utility ranking, recovering the same subsets as U. The coverage-based completion in $C + G$ can produce different subsets at this budget. Figure 4 helps explain why the angular margin has a clear effect despite changing the feasible pool by just 4.1% on `peg` and 3.6% on `stackcube`. The excluded candidates are nearly orthogonal to target return, so their residual components can yield large coverage gains while providing little support for target success. Removing these misleading directions keeps coverage focused on performance-aligned behavior.

6 Discussions

We answer the questions posed in Section 5 and discuss their implications for robot data curation.

Q1: Curation across domain gaps. Our study shows that a single TUCO curation mechanism remains effective from single-simulator selection to Sim-to-Sim and Sim-to-Real co-training. This finding yields two insights into robust curation. First, demonstrations should be evaluated through

Table 2: Component ablations in sim-to-sim co-training. We report success rates (%).

Budget (%)	U	C+G	U+C	TUCO
`Peg`				
5	53.8±5.8	54.8±4.1	**57.0±3.9**	56.7±3.3
10	58.2±5.0	59.6±2.1	58.6±5.7	**61.4±2.6**
15	59.4±2.4	61.9±4.8	62.6±4.1	**63.3±3.1**
25	68.1±4.0	66.9±0.8	62.3±5.1	**71.6±2.3**
50	**70.9±4.0**	68.7±1.2	62.0±7.0	**70.9±4.0**
75	**67.4±1.3**	**67.4±2.1**	**67.4±1.3**	**67.4±1.3**
`StackCube`				
5	37.1±3.3	36.1±5.2	39.3±5.4	**39.9±4.0**
10	36.1±2.2	37.1±4.7	36.0±0.9	**42.3±4.4**
15	39.5±4.9	39.4±1.1	40.0±0.3	**43.2±2.8**
25	40.9±2.5	40.9±2.1	**42.3±1.3**	42.1±2.1
50	44.0±1.5	41.4±3.5	43.6±2.9	**45.5±1.5**
75	**42.6±2.9**	40.9±3.3	**42.6±2.9**	**42.6±2.9**
`CupCake`				
5	50.4±1.8	51.6±3.0	51.4±1.8	**52.5±3.3**
10	46.7±2.1	48.5±3.6	50.6±1.3	**52.2±4.5**
15	53.0±2.3	47.7±2.6	53.1±2.9	**54.1±2.6**
25	54.3±1.7	49.3±2.1	50.7±2.4	**54.5±3.3**
50	52.2±4.0	**52.4±2.5**	52.2±4.0	52.2±4.0
75	**49.5±3.5**	45.5±2.4	**49.5±3.5**	**49.5±3.5**

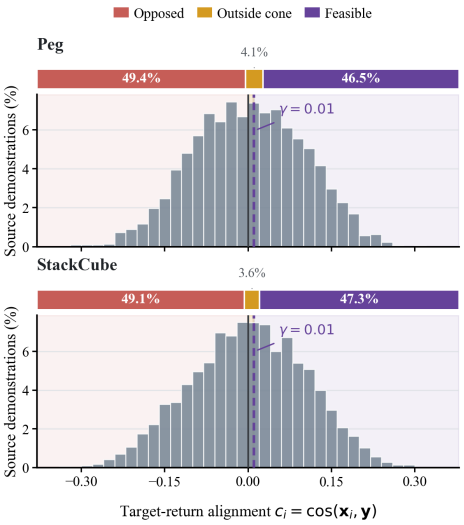

Figure 4: Cone partitions for `Peg` and `StackCube`. Bars show region proportions; histograms show alignment c_i at $\gamma = 0.01$.

their effects on target-policy closed-loop performance, since criteria based on latent structure or proxy objectives can capture domain-specific variation whose relevance changes across environments. Second, curation should be treated as set construction because a demonstration's value depends on the target data and the behaviors already represented by the selected subset. TUCO realizes both principles by measuring trajectory-level utility and set-level coverage from shared target-domain scoring rollouts, which explains its consistency as the domain gap changes.

Q2: Active curation for sim-to-real co-training. To the best of our knowledge, our study provides the first systematic evidence that active curation benefits both Sim-to-Sim and Sim-to-Real co-training, and leads to a new insight: co-training should optimize which source demonstrations to mix in addition to how much source data to mix. Curated subsets outperform co-training with the full source pool, while random subsets of the same size do not recover the same benefit. With the target demonstrations also held fixed, this comparison identifies source-data composition as a consequential choice beyond the relative amounts of source and target data. Active curation therefore helps convert a broad simulation pool into target-relevant supervision by selecting useful and complementary demonstrations.

7 Conclusions

In this work, we studied simulation data curation for robot policy co-training and presented a systematic evaluation of its value in Sim-to-Real co-training. We introduced TUCO, a set-level curation framework that grounds source-data selection in target-policy closed-loop behavior. We used influence functions to trace each source demonstration to its effects on target-domain scoring rollouts, then decomposed these effects to measure trajectory-level utility and set-level coverage within a shared closed-loop space. We further developed a performance-aligned subset optimizer that combines both signals to suppress redundancy and retain complementary demonstrations. Across RoboMimic and OmniReset, TUCO achieves the best or tied-best performance among the compared methods across Single-Simulator, Sim-to-Sim, and Sim-to-Real settings. More importantly, our experiments show that active curation improves cross-domain co-training over using the full source pool and outperforms random selection at matched data budgets, establishing the value of choosing which simulation demonstrations to mix.

AI USE STATEMENT

In this work, we used generative AI tools for no tasks requiring mandatory disclosure. We have not used generative AI tools for any other tasks requiring mandatory disclosure, and the remaining required-disclosure tasks are not applicable to this work. Additionally, we used generative AI tools to identify and survey relevant literature, proofread the manuscript to improve its readability and language, and refactor and improve the structure of the software code. We have reviewed all AI-assisted work. All references and literature-related information suggested by generative AI were independently verified by the authors against the original sources. All AI-assisted textual revisions were reviewed to ensure that they preserved the authors' intended meaning and did not alter the scientific claims. All AI-assisted code modifications were manually inspected and tested by the authors to verify their correctness and to ensure that they did not change the intended algorithms or experimental procedures. We take responsibility for the final content of this work, including text, claims, or artifacts produced with the aid of generative AI.

ETHICS STATEMENT

This study does not analyze human participants or personal data. Real-robot experiments were conducted in a controlled laboratory environment under human supervision. Although TUCO improves data selection for robot policy learning, it does not provide a safety guarantee. Policies should undergo task-specific safety evaluation before deployment in unconstrained environments.

REPRODUCIBILITY STATEMENT

The main paper provides the problem formulation, complete TUCO objective, and selection algorithm. Appendix B contains the supporting derivations and implementation details, while Appendix C documents the datasets, policy training, baseline adaptations, evaluation protocols. Code, configurations, and data-processing scripts will be provided in the supplementary material.

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 *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 2907–2932. PMLR, 27–30 September 2025.

Brenna D. Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

Aarav Bedi. What demonstration curation metrics do to your policy. *arXiv preprint arXiv:2606.10229*, 2026.

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

Paweł Budzianowski, Emilia Wiśnios, Michał Tyrolski, Gracjan Góral, Igor Kulakov, Viktor Petrenko, and Krzysztof Walas. OpenGVL – benchmarking visual temporal progress for data curation. *arXiv preprint arXiv:2509.17321*, 2025.

Annie S. Chen, Alec M. Lessing, Yuejiang Liu, and Chelsea Finn. Curating demonstrations using online experience. *arXiv preprint arXiv:2503.03707*, 2025.

Shuo Cheng, Liqian Ma, Zhenyang Chen, Ajay Mandlekar, Caelan Garrett, and Danfei Xu. Generalizable domain adaptation for sim-and-real policy co-training. In *Advances in Neural Information Processing Systems*, volume 38, 2025.

Shivin Dass, Alaa Khaddaj, Logan Engstrom, Aleksander Madry, Andrew Ilyas, and Roberto Martín-Martín. DataMIL: Selecting data for robot imitation learning with datamodels. In *International Conference on Learning Representations*, 2026.

Maximilian Du, Suraj Nair, Dorsa Sadigh, and Chelsea Finn. Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets. In *Proceedings of Robotics: Science and Systems*, Daegu, Republic of Korea, July 2023.

Kaipeng Fang, Weiqing Liang, Yuyang Li, Ji Zhang, Pengpeng Zeng, Lianli Gao, Jingkuan Song, and Heng Tao Shen. Sim-and-human co-training for data-efficient and generalizable robotic manipulation. *arXiv preprint arXiv:2601.19406*, 2026.

Lan Feng, Fan Nie, Yuejiang Liu, and Alexandre Alahi. TAROT: Targeted data selection via optimal transport. In *Proceedings of the 42nd International Conference on Machine Learning*, volume 267 of *Proceedings of Machine Learning Research*, pp. 16837–16852. PMLR, 2025.

Kanishk Gandhi, Siddharth Karamcheti, Madeline Liao, and Dorsa Sadigh. Eliciting compatible demonstrations for multi-human imitation learning. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 1981–1991. PMLR, 2023.

Joey Hejna, Chethan Bhateja, Yichen Jiang, Karl Pertsch, and Dorsa Sadigh. Re-Mix: Optimizing data mixtures for large scale imitation learning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 145–164. PMLR, 6–9 November 2025a.

Joey Hejna, Suvir Mirchandani, Ashwin Balakrishna, Annie Xie, Ayzaan Wahid, Jonathan Tompson, Pannag R. Sanketi, Dhruv Shah, Coline Manon Devin, and Dorsa Sadigh. Robot data curation with mutual information estimators. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025b.

Xiaogang Jia, Denis Blessing, Xinkai Jiang, Moritz Reuss, Atalay Donat, Rudolf Lioutikov, and Gerhard Neumann. Towards diverse behaviors: A benchmark for imitation learning with human demonstrations. In *International Conference on Learning Representations*, 2024.

Seungku Kim, Suhyeok Jang, Byungjun Yoon, Dongyoung Kim, John Won, and Jinwoo Shin. RoboCurate: Harnessing diversity with action-verified neural trajectory for robot learning. *arXiv preprint arXiv:2602.18742*, 2026.

Pang Wei Koh and Percy Liang. Understanding black-box predictions via influence functions. In *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 1885–1894. PMLR, 2017.

Sachit Kuhar, Shuo Cheng, Shivang Chopra, Matthew Bronars, and Danfei Xu. Learning to discern: Imitating heterogeneous human demonstrations with preference and representation learning. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1437–1449. PMLR, 2023.

Soham Kulkarni, Raayan Dhar, and Yuchen Cui. Learning from the best: Smoothness-driven metrics for data quality in imitation learning. *arXiv preprint arXiv:2604.23000*, 2026.

Sateesh Kumar, Shivin Dass, Georgios Pavlakos, and Roberto Martín-Martín. COLLAGE: Adaptive fusion-based retrieval for augmented policy learning. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 4607–4624. PMLR, 2025.

Shreyas Kumar and Ravi Prakash. PACER: Progress-aligned curation for error-resilient imitation learning. In *CoRL Workshop on Making Sense of Data in Robotics: Composition, Curation, and Interpretability at Scale*, 2025.

Haeone Lee, Taywon Min, Junsu Kim, Sinjae Kang, Fangchen Liu, Lerrel Pinto, and Kimin Lee. Quality over quantity: Demonstration curation via influence functions for data-centric robot learning. *arXiv preprint arXiv:2603.09056*, 2026.

Yu Lei, Minghuan Liu, Abhiram Maddukuri, Zhenyu Jiang, and Yuke Zhu. A mechanistic analysis of sim-and-real co-training in generative robot policies. *arXiv preprint arXiv:2604.13645*, 2026.

Haozhuo Li, Yuchen Cui, and Dorsa Sadigh. How to train your robots? the impact of demonstration modality on imitation learning. In *IEEE International Conference on Robotics and Automation*, pp. 1113–1120, 2025.

Fanqi Lin, Yingdong Hu, Pingyue Sheng, Chuan Wen, Jiacheng You, and Yang Gao. Data scaling laws in imitation learning for robotic manipulation. In *International Conference on Learning Representations*, 2025a.

Fanqi Lin, Kushal Arora, Jean Mercat, Haruki Nishimura, Paarth Shah, Chen Xu, Mengchao Zhang, Mark Zolotas, Maya Angeles, Owen Pfannenstiehl, Andrew Beaulieu, and Jose Barreiros. A systematic study of data modalities and strategies for co-training large behavior models for robot manipulation. *arXiv preprint arXiv:2602.01067*, 2026a.

Li-Heng Lin, Yuchen Cui, Amber Xie, Tianyu Hua, and Dorsa Sadigh. FlowRetrieval: Flow-guided data retrieval for few-shot imitation learning. In *Proceedings of The 8th Conference on Robot Learning*, volume 270 of *Proceedings of Machine Learning Research*, pp. 4084–4099. PMLR, 2025b.

Nie Lin, Takehiko Ohkawa, Sijin Chen, Ruoshi Wen, Zhuohang Li, Liqun Huang, Zhengming Zhu, Yiming Bao, Yunfei Li, Minjie Cai, Xiao Ma, Wei Xu, and Yoichi Sato. SiMDex: Mining similar egocentric videos for cross-embodiment dexterous manipulation. *arXiv preprint arXiv:2608.04196*, 2026b.

Abhiram Maddukuri, Zhenyu Jiang, Lawrence Yunliang Chen, Soroush Nasiriany, Yuqi Xie, Yu Fang, Wenqi Huang, Zu Wang, Zhenjia Xu, Nikita Chernyadev, Scott Reed, Ken Goldberg, Ajay Mandlekar, Linxi Fan, and Yuke Zhu. Sim-and-real co-training: A simple recipe for vision-based robotic manipulation. In *Proceedings of Robotics: Science and Systems*, Los Angeles, CA, USA, June 2025.

Ajay Mandlekar, Danfei Xu, Josiah Wong, Soroush Nasiriany, Chen Wang, Rohun Kulkarni, Li Fei-Fei, Silvio Savarese, Yuke Zhu, and Roberto Martín-Martín. What matters in learning from offline human demonstrations for robot manipulation. In *Proceedings of the 5th Conference on Robot Learning*, volume 164 of *Proceedings of Machine Learning Research*, pp. 1678–1690. PMLR, 2022.

Ajay Mandlekar, Soroush Nasiriany, Bowen Wen, Iretiayo Akinola, Yashraj Narang, Linxi Fan, Yuke Zhu, and Dieter Fox. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *Proceedings of The 7th Conference on Robot Learning*, volume 229 of *Proceedings of Machine Learning Research*, pp. 1820–1864. PMLR, 2023.

Marius Memmel, Jacob Berg, Bingqing Chen, Abhishek Gupta, and Jonathan Francis. STRAP: Robot sub-trajectory retrieval for augmented policy learning. In *International Conference on Learning Representations*, 2025.

Kana Miyamoto, Kanata Suzuki, and Tetsuya Ogata. How to utilize failure demo data?: Effective data selection for imitation learning using distribution differences in attention mechanism. *arXiv preprint arXiv:2605.07560*, 2026.

Soroush Nasiriany, Tian Gao, Ajay Mandlekar, and Yuke Zhu. Learning and retrieval from prior data for skill-based imitation learning. In *Proceedings of The 6th Conference on Robot Learning*, volume 205 of *Proceedings of Machine Learning Research*, pp. 2181–2204. PMLR, 2023.

Soroush Nasiriany, Abhiram Maddukuri, Lance Zhang, Adeet Parikh, Aaron Lo, Abhishek Joshi, Ajay Mandlekar, and Yuke Zhu. RoboCasa: Large-scale simulation of household tasks for generalist robots. In *Proceedings of Robotics: Science and Systems*, Delft, Netherlands, July 2024.

Takayuki Osa, Joni Pajarinen, Gerhard Neumann, J. Andrew Bagnell, Pieter Abbeel, and Jan Peters. An algorithmic perspective on imitation learning. *Foundations and Trends in Robotics*, 7(1–2): 1–179, 2018.

Hrishikesh Sathyanarayan, Victor Vantilborgh, and Ian Abraham. Quality over quantity: Curating contact-based robot datasets improves learning. *arXiv preprint arXiv:2510.18137*, 2025.

Vaibhav Saxena, Matthew Bronars, Nadun Ranawaka Arachchige, Kuancheng Wang, Woo Chul Shin, Soroush Nasiriany, Ajay Mandlekar, and Danfei Xu. What matters in learning from large-scale datasets for robot manipulation. In *International Conference on Learning Representations*, 2025.

Liangzhi Shi, Shuaihang Chen, Feng Gao, Yinuo Chen, Kang Chen, Tonghe Zhang, Hongzhi Zang, Jiakai Zhou, Weinan Zhang, Chao Yu, and Yu Wang. Beyond imitation: Reinforcement learning-based sim-real co-training for VLA models. *arXiv preprint arXiv:2602.12628*, 2026.

Modi Shi, Li Chen, Jin Chen, Yuxiang Lu, Chiming Liu, Guanghui Ren, Ping Luo, Di Huang, Maoqing Yao, and Hongyang Li. Is diversity all you need for scalable robotic manipulation? *arXiv preprint arXiv:2507.06219*, 2025.

Qie Sima, Wei Xue, and Yike Guo. ProMix: Learning optimal data mixtures for robotic imitation via proxy-reference distillation. In *ICRA Workshop on From Data to Decisions: VLA Pipelines for Real Robots*, 2026.

Sreevardhan Sirigiri, Nathan Samuel de Lara, Christopher Agia, Florian Shkurti, and Fabio Ramos. Diversity you can actually measure: A fast, model-free diversity metric for robotics datasets. *arXiv preprint arXiv:2603.11634*, 2026.

Noushad Sojib, Sajay Arthanat, and Momotaz Begum. An efficient metric for data quality measurement in imitation learning. *arXiv preprint arXiv:2605.01544*, 2026.

Adam Wei, Abhinav Agarwal, Boyuan Chen, Rohan Bosworth, Nicholas Pfaff, and Russ Tedrake. Empirical analysis of sim-and-real cotraining of diffusion policies for planar pushing from pixels. In *IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 5625–5632, 2025.

Adam Wei, Nicholas Pfaff, Thomas Cohn, Arif Kerem Dayı, Constantinos Daskalakis, Giannis Daras, and Russ Tedrake. Ambient diffusion policy: Imitation learning from suboptimal data in robotics. *arXiv preprint arXiv:2606.12365*, 2026.

Changti Wu, Bin Yu, Zhaolong Shen, Shijie Lian, Xiaopeng Lin, Cong Huang, Zhirui Zhang, Lei Zhang, and Kai Chen. SIEVE: Structure-aware data selection for imitation learning with VLA models. *arXiv preprint arXiv:2607.06442*, 2026.

Jiahao Xiao, Bowen Yan, Jianbo Zhang, Jia Wang, Chunyi Li, Zhengxue Cheng, and Guangtao Zhai. Data assessment for embodied intelligence. *arXiv preprint arXiv:2511.09119*, 2025.

Amber Xie, Rahul Chand, Dorsa Sadigh, and Joey Hejna. Data retrieval with importance weights for few-shot imitation learning. In *Proceedings of The 9th Conference on Robot Learning*, volume 305 of *Proceedings of Machine Learning Research*, pp. 1–16. PMLR, 2025.

Youguang Xing, Xu Luo, Junlin Xie, Lianli Gao, Hengtao Shen, and Jingkuan Song. Shortcut learning in generalist robot policies: The role of dataset diversity and fragmentation. *arXiv preprint arXiv:2508.06426*, 2025.

Tao Xu, Jiaxin Wang, Runhao Zhang, Jiayi Guan, Xianchao Zeng, Weixi Song, Xinyu Zhou, Zhetao Chen, Guang Chen, and Yong-Lu Li. ATHENA: Accelerated multi-task heterogeneous influence functions for robot data curation. *arXiv preprint arXiv:2606.16208*, 2026.

Jonathan Yang, Chelsea Finn, and Dorsa Sadigh. Invariance co-training for robot visual generalization. *arXiv preprint arXiv:2512.05230*, 2025.

Jonathan Yang, Chelsea Finn, and Dorsa Sadigh. Data analogies enable efficient cross-embodiment transfer. *arXiv preprint arXiv:2603.06450*, 2026a.

Licheng Yang, Lingfeng Qian, Fei Zheng, Yonghao He, Wei Sui, Shuangshuang Li, and Hu Su. Improving robotic imitation learning via trajectory standardization. *arXiv preprint arXiv:2606.22907*, 2026b.

Patrick Yin, Tyler Westenbroek, Zhengyu Zhang, Ignacio Dagnino, Eeshani Shilamkar, Numfor Mbiziwo-Tiapo, Simran Bagaria, Xinlei Liu, Galen Mullins, Andrey Kolobov, and Abhishek Gupta. Emergent dexterity via diverse resets and large-scale reinforcement learning. In *International Conference on Learning Representations*, 2026.

Bin Yu, Shijie Lian, Xiaopeng Lin, Zhaolong Shen, Yuliang Wei, Changti Wu, Hang Yuan, Haishan Liu, Bailing Wang, Cong Huang, and Kai Chen. FrameSkip: Learning from fewer but more informative frames in VLA training. *arXiv preprint arXiv:2605.13757*, 2026a.

Justin Yu, Andrew Goldberg, Kavish Kondap, Karim El-Refai, Ethan Ransing, Qianzhong Chen, Mac Schwager, Fred Shentu, Philipp Wu, and Ken Goldberg. WARP-RM: A warp-augmented relative progress reward model for data curation. *arXiv preprint arXiv:2606.28320*, 2026b.

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*, 2026.

Mengfei Zhao, Dihong Huang, Yikai Tang, Peihao Li, Mingxuan Yan, Ruiqi Zhuang, Yanjia Huang, Jie Wang, Hai Zhai, Tony Zhou, Rui Zhang, Zhexi Luo, Yuchen Huang, Jianfei Yang, and Jiachen Li. Axis: A growable community-driven data engine for scalable robot manipulation, 2026. URL https://arxiv.org/abs/2607.21588.

APPENDIX CONTENTS

A Extended Related Work **16**
A.1 Data Curation for Robot Learning 16
A.2 Cross-Domain Robot Co-Training 16

B Theoretical Details **16**
B.1 Closed-Loop Performance Influence 16
B.2 Practical Influence Estimation 17
B.3 Influence Decomposition and Subset Geometry 18
B.4 Computational Complexity 19

C Additional Experimental Details **19**
C.1 Tasks and Datasets 19
C.2 Simulation Policy Training Details 20
C.3 Real-World Policy Training Details 20
C.4 Baseline Implementations 22
C.5 Evaluation Protocol 24
C.6 Influence-Space Visualization 24

D Extended Experiments **24**
D.1 Hyperparameter Sensitivity 24
D.2 Robustness to Unseen Lighting Conditions. 26

E Extended Discussions **26**
E.1 Insights into Robust Curation across Domain Gaps 26
E.2 Source Composition as a Co-Training Design Variable 27
E.3 Limitations 27

A Extended Related Work

A.1 Data Curation for Robot Learning

Robot data curation methods can be categorized into three groups. **Quality-based approaches** estimate value from demonstration properties. Existing studies analyze consistency and heterogeneity (Mandlekar et al., 2022; Belkhale et al., 2023), demonstrator compatibility (Gandhi et al., 2023; Kuhar et al., 2023), and collection modality (Li et al., 2025). Scalable criteria use state-action dependence (Hejna et al., 2025b) or learned task progress (Zhang et al., 2026; Budzianowski et al., 2025), while temporal and motion cues offer policy-free alternatives (Kumar & Prakash, 2025; Sojib et al., 2026; Kulkarni et al., 2026). Recent extensions operate on frames and action segments (Yu et al., 2026a;b), adapt temporal sampling (Yang et al., 2026b), and evaluate failure data (Miyamoto et al., 2026). Other criteria verify synthetic actions (Kim et al., 2026), focus on contact-sensitive phases (Sathyanarayan et al., 2025), or assess the behavioral consequences of quality metrics (Bedi, 2026). **Utility-based approaches** condition value on target data or policy behavior. Retrieval methods match skill and state-action structure (Nasiriany et al., 2023; Du et al., 2023), often using visual or motion features for cross-task transfer (Lin et al., 2025b; Memmel et al., 2025). Density correction (Xie et al., 2025), cue fusion (Kumar et al., 2025), and embodiment alignment (Lin et al., 2026b) support heterogeneous sources. Policy-conditioned methods use rollout outcomes (Chen et al., 2025) or learned datamodels (Dass et al., 2026). Influence functions (Koh & Liang, 2017) connect training data to imitation loss (Lee et al., 2026) and closed-loop return (Agia et al., 2025; Xu et al., 2026), while mixture optimization estimates utility over source domains (Hejna et al., 2025a; Sima et al., 2026). **Diversity-based approaches** suppress representation or trajectory redundancy (Zhang et al., 2026; Sirigiri et al., 2026). Distribution matching (Feng et al., 2025) and primitive composition (Wu et al., 2026) pursue broader coverage. This set-level view is supported by studies of behavioral multimodality (Jia et al., 2024), task and embodiment diversity (Shi et al., 2025), and dataset scaling (Lin et al., 2025a; Saxena et al., 2025). Complementary analyses examine learnability (Xiao et al., 2025) and fragmentation (Xing et al., 2025) in generalist robot data.

A.2 Cross-Domain Robot Co-Training

Simulation platforms provide scalable source experience for robot policy learning (Mandlekar et al., 2023; Nasiriany et al., 2024). Cross-domain co-training can be organized by how this experience is composed, aligned, and used during optimization. **Composition-oriented studies** examine simulation scale and fixed sim-to-real mixing ratios (Wei et al., 2025; Maddukuri et al., 2025), with heterogeneous variants combining predefined modalities or embodiments (Lin et al., 2026a; Yang et al., 2026a). **Alignment-oriented methods** improve source compatibility through observation-action alignment (Cheng et al., 2025), visual invariance (Yang et al., 2025), or complementary simulation and human priors (Fang et al., 2026). Mechanistic analyses further connect co-training gains to representation alignment and domain-dependent action weighting (Lei et al., 2026). **Utilization-oriented methods** learn weights over predefined source domains (Hejna et al., 2025a; Sima et al., 2026), schedule lower-quality data during diffusion training (Wei et al., 2026), or combine simulation reinforcement learning with real demonstrations (Shi et al., 2026). These approaches establish the importance of source composition and training strategy. However, sim-to-real co-training lacks a curation mechanism that actively selects simulation demonstrations according to their contribution to real-world policy learning.

B Theoretical Details

Roadmap. We derive closed-loop return influence for a target-trained base policy in Appendix B.1 and describe practical scoring with both newly collected and recorded rollouts in Appendix B.2. Appendix B.3 establishes the decomposition and coverage geometry, followed by a complexity analysis in Appendix B.4.

B.1 Closed-Loop Performance Influence

We justify the return-influence interpretation of the rollout-wise profiles and utility scores in Section 4.2 for a base policy trained on the target data. We use the standard influence-function re-

sult (Koh & Liang, 2017) and CUPID's closed-loop attribution argument (Agia et al., 2025) in our source-to-target notation.

Conditions. We assume that θ_0 is an interior local minimizer of $\mathcal{L}_{\mathrm{t}}$ with nonsingular Hessian $\mathbf{H}_0$, and that both $\mathcal{L}_{\mathrm{t}}$ and $\ell_\xi(\xi_i^{\mathrm{s}} \mid \theta)$ are twice continuously differentiable near θ_0. The target environment and return function are independent of θ. The rollout density is differentiable with locally fixed support, and differentiation can be exchanged with integration. The return and return-weighted rollout score are integrable.

Theorem 1 (Return-Weighted Performance Influence)**.** *Under the conditions above, let $\theta_{\epsilon,i}$ be the local minimizer of $\mathcal{L}_{\mathrm{t}}(\theta) + \epsilon\ell_\xi(\xi_i^{\mathrm{s}} \mid \theta)$ continuing from $\theta_{0,i} = \theta_0$. Define $\mathbf{g}_i := \nabla_\theta \ell_\xi(\xi_i^{\mathrm{s}} \mid \theta_0)$ and $\mathbf{h}_r := \nabla_\theta \log p_{\mathrm{t}}(\tau_r \mid \pi_\theta)|_{\theta=\theta_0}$. For a target rollout in the support of the base policy,*

$$\begin{aligned} X_{ri} &:= \partial_\epsilon \log p_{\mathrm{t}}(\tau_r \mid \pi_{\theta_{\epsilon,i}})\big|_{\epsilon=0} \\ &= -\mathbf{h}_r^\top \mathbf{H}_0^{-1} \mathbf{g}_i. \end{aligned} \tag{5}$$

With $y_r := R_{\mathrm{t}}(\tau_r)$, its return-weighted expectation gives

$$\begin{aligned} \mathcal{I}_i^{\mathrm{perf}} &:= \partial_\epsilon J_{\mathrm{t}}(\theta_{\epsilon,i})|_{\epsilon=0} \\ &= \mathbb{E}_{\tau_r \sim p_{\mathrm{t}}(\cdot|\pi_{\theta_0})}[y_r X_{ri}]. \end{aligned} \tag{6}$$

Proof. Differentiating the stationarity condition for the perturbed objective gives the standard parameter response (Koh & Liang, 2017):

$$\partial_\epsilon \theta_{\epsilon,i}|_{\epsilon=0} = -\mathbf{H}_0^{-1} \mathbf{g}_i. \tag{7}$$

The environment factors in the rollout likelihood are independent of θ, so

$$\mathbf{h}_r = \sum_{h=0}^{H_r-1} \nabla_\theta \log \pi_\theta(a_{r,h} \mid o_{r,h})|_{\theta=\theta_0}. \tag{8}$$

Applying the chain rule to the log likelihood of a fixed rollout then yields Equation 5.

For expected return, the likelihood-ratio identity gives $\nabla_\theta J_{\mathrm{t}}(\theta_0) = \mathbb{E}_{\tau_r}[y_r \mathbf{h}_r]$. Combining it with the parameter response yields

$$\begin{aligned} \mathcal{I}_i^{\mathrm{perf}} &= \nabla_\theta J_{\mathrm{t}}(\theta_0)^\top \, \partial_\epsilon \theta_{\epsilon,i}|_{\epsilon=0} \\ &= -\mathbb{E}_{\tau_r}[y_r \mathbf{h}_r]^\top \mathbf{H}_0^{-1} \mathbf{g}_i \\ &= \mathbb{E}_{\tau_r}[y_r X_{ri}], \end{aligned} \tag{9}$$

where $\mathbf{H}_0$ and $\mathbf{g}_i$ are fixed under the rollout expectation. □

By Theorem 1, $q_i/M = M^{-1} \sum_{r=1}^{M} y_r X_{ri}$ is an unbiased estimator of local return influence when the theorem's conditions hold, profile entries are exact, and the scoring rollouts are independently sampled from $p_{\mathrm{t}}(\cdot \mid \pi_{\theta_0})$. With returns $+1$ for success and -1 for failure, $J_{\mathrm{t}} = 2P_{\mathrm{success}} - 1$, so $q_i/(2M)$ estimates the corresponding first-order change in success probability. The next subsection describes the practical estimator and the interpretation of the success-only feedback used in Sim-to-Real.

B.2 Practical Influence Estimation

We follow CUPID (Agia et al., 2025) to estimate influence profiles using projected loss gradients. We evaluate all gradients at the fixed base policy θ_0 and use the same target-domain scoring rollouts for every source candidate. In the simulated settings, the base policy is trained on target demonstrations. For Sim-to-Real, it is pretrained on simulation data and supplies recorded successful real-world rollouts.

Trajectory aggregation. Let $\mathbf{e}_{r,h} := \nabla_\theta \ell(o_{r,h}, a_{r,h} \mid \theta_0)$ and $\mathbf{g}_{i,k} := \nabla_\theta \ell(o^{\mathrm{s}}_{i,k}, a^{\mathrm{s}}_{i,k} \mid \theta_0)$ denote target-rollout and source-demonstration loss gradients. For a negative log-likelihood loss, $\mathbf{h}_r = -\sum_h \mathbf{e}_{r,h}$ and $\mathbf{g}_i = \sum_k \mathbf{g}_{i,k}$. Thus,

$$X_{ri} = \sum_{h=0}^{H_r-1} \sum_{k=0}^{T_i-1} \mathbf{e}_{r,h}^\top \mathbf{H}_0^{-1} \mathbf{g}_{i,k}. \tag{10}$$

For a deterministic MLP, we use mean-squared behavior-cloning loss as a surrogate. This loss corresponds to Gaussian negative log likelihood up to a positive scale and an additive constant when the action covariance is fixed and isotropic.

Projected estimator. Let $\mathbf{P} \in \mathbb{R}^{d_\theta \times d}$ be a shared random projection from the d_θ policy parameters to d dimensions. We collect the projected gradients of source training examples as rows of $\widetilde{\mathbf{G}} \in \mathbb{R}^{L_{\mathrm{s}} \times d}$, where L_{s} counts individual transitions for the state policy or observation–action windows for Diffusion Policy, and define

$$\widetilde{\mathbf{C}} := \widetilde{\mathbf{G}}^\top \widetilde{\mathbf{G}} + \eta \mathbf{I}_d, \qquad \eta \geq 0. \tag{11}$$

We require $\widetilde{\mathbf{C}}$ to be invertible. This source-gradient Gram matrix provides the curvature approximation used for scoring at the frozen base policy. In the exact target-trained derivation, $\mathbf{H}_0$ is the Hessian of the target training loss. With $\widetilde{\mathbf{e}}_r := \sum_h \mathbf{P}^\top \mathbf{e}_{r,h}$ and $\widetilde{\mathbf{g}}_i := \sum_k \mathbf{P}^\top \mathbf{g}_{i,k}$, the estimated profile entry is

$$\widehat{X}_{ri} = \widetilde{\mathbf{e}}_r^\top \widetilde{\mathbf{C}}^{-1} \widetilde{\mathbf{g}}_i. \tag{12}$$

Bilinearity allows trajectory gradients to be summed before scoring, preserving the frame-pair aggregation in Equation 10 under this curvature approximation. We use these estimated profiles as $\mathbf{X}$ in TUCO's decomposition and subset selection.

B.3 Influence Decomposition and Subset Geometry

We establish the utility-preserving decomposition in Section 4.3 and the volume interpretation and marginal gain of the coverage criterion in Section 4.4. The following identities hold for any fixed influence profiles and $\mathbf{y} \neq \mathbf{0}$, including the estimated profiles constructed from recorded successful rollouts.

Decomposition and performance alignment. Recall $\mathbf{x}_i = u_i \widehat{\mathbf{y}} + \mathbf{z}_i$ from Equation 1. Since $\mathbf{z}_i$ is orthogonal to the unit vector $\widehat{\mathbf{y}}$,

$$\begin{aligned} \widehat{\mathbf{y}}^\top \mathbf{z}_i &= 0, \\ q_i &= \|\mathbf{y}\|_2 u_i, \\ \|\mathbf{x}_i\|_2^2 &= u_i^2 + \|\mathbf{z}_i\|_2^2. \end{aligned} \tag{13}$$

The decomposition therefore preserves utility. When $q_i = q_j$, we have $\mathbf{x}_i - \mathbf{x}_j = \mathbf{z}_i - \mathbf{z}_j$, so the residuals retain all differences between profiles with equal utility. For $q_i > 0$, the alignment constraint in Equation 2 becomes

$$\frac{u_i}{\sqrt{u_i^2 + \|\mathbf{z}_i\|_2^2}} \geq \gamma. \tag{14}$$

For $\gamma \in (0, 1)$, this is equivalent to $\|\mathbf{z}_i\|_2 \leq u_i \sqrt{\gamma^{-2} - 1}$. The cone thus bounds residual magnitude relative to the positive return-aligned component before optimizing directional coverage. At $\gamma = 0$, it reduces to the positive-utility half-space.

For success-only feedback, $\mathbf{y} = \mathbf{1} \in \mathbb{R}^M$ and $\widehat{\mathbf{y}} = \mathbf{1}/\sqrt{M}$. The decomposition becomes

$$q_i = \sum_{r=1}^{M} X_{ri}, \qquad u_i = \frac{q_i}{\sqrt{M}}, \qquad \mathbf{z}_i = \mathbf{x}_i - \frac{q_i}{M}\mathbf{1}. \tag{15}$$

Thus, utility measures aggregate support across the successful scoring rollouts, while the residual records variation around that profile's mean influence. Even with identical success labels, the residuals retain differences across executions that can support complementary coverage. The same cone constraint and subset optimizer apply to these profiles.

Spanning-volume interpretation. For a nonempty feasible pool, use the positive weights $w_i = q_i/\kappa_{\mathcal{F}}$ from the main text. Normalize nonzero residuals to $\mathbf{v}_i = \mathbf{z}_i/\|\mathbf{z}_i\|_2$ and set $\mathbf{v}_i = \mathbf{0}$ otherwise. For $S \subseteq \mathcal{F}_\gamma$, define

$$\begin{aligned} \mathbf{V}_S &:= [\sqrt{\rho w_i}\mathbf{v}_i]_{i\in S}, \\ \mathbf{M}(S) &:= \mathbf{I}_M + \mathbf{V}_S\mathbf{V}_S^\top. \end{aligned} \tag{16}$$

Because $\mathbf{M}(S) \succeq \mathbf{I}_M$, the coverage score $G(S) = \log\det\mathbf{M}(S)$ is well defined for every subset, including the empty set. Sylvester's determinant identity and the principal-minor expansion give

$$\begin{aligned} \exp G(S) &= \det\big(\mathbf{I}_{|S|} + \mathbf{V}_S^\top\mathbf{V}_S\big) \\ &= \sum_{T\subseteq S} \rho^{|T|} \left(\prod_{i\in T} w_i\right) \det\big([\mathbf{v}_i^\top\mathbf{v}_j]_{i,j\in T}\big)\,. \end{aligned} \tag{17}$$

The empty-set term is one. Each Gram determinant equals the squared parallelotope volume spanned by its residual vectors and vanishes for linearly dependent vectors. Thus, $\exp G(S)$ aggregates utility-weighted squared spanning volumes in the subspace orthogonal to $\mathbf{y}$. This gives a precise interpretation of the coverage intuition in the main text. The outer product $\mathbf{v}_i\mathbf{v}_i^\top$ treats opposite residual directions as the same span. A zero residual contributes no coverage while remaining eligible if it satisfies the cone constraint.

Marginal coverage. Adding $i \in \mathcal{F}_\gamma \setminus S$ updates the coverage matrix by $\rho w_i\mathbf{v}_i\mathbf{v}_i^\top$. Applying the matrix determinant lemma gives

$$\begin{aligned} \Delta_i(S) &= \log\frac{\det(\mathbf{M}(S) + \rho w_i\mathbf{v}_i\mathbf{v}_i^\top)}{\det\mathbf{M}(S)} \\ &= \log\big(1 + \rho w_i\mathbf{v}_i^\top\mathbf{M}(S)^{-1}\mathbf{v}_i\big)\,. \end{aligned} \tag{18}$$

This is the marginal gain used in Algorithm 1. To see how it accounts for existing coverage, write $\mathbf{M}(S) = \mathbf{Q}\,\mathrm{diag}(d_1,\ldots,d_M)\mathbf{Q}^\top$ in an orthonormal eigenbasis. Then

$$\mathbf{v}_i^\top\mathbf{M}(S)^{-1}\mathbf{v}_i = \sum_{k=1}^{M} \frac{(\mathbf{Q}_{:,k}^\top\mathbf{v}_i)^2}{d_k}, \qquad d_k \geq 1. \tag{19}$$

A larger d_k indicates greater weighted support along that direction in S. Its inverse reduces the contribution of an already covered direction, favoring additional coverage for candidates with comparable utility. For $S \subseteq T \subseteq \mathcal{F}_\gamma$, we have $\mathbf{M}(T) \succeq \mathbf{M}(S) \succ 0$, hence $\mathbf{M}(T)^{-1} \preceq \mathbf{M}(S)^{-1}$. Equation 18 therefore implies $0 \leq \Delta_i(T) \leq \Delta_i(S)$ for $i \in \mathcal{F}_\gamma \setminus T$.

Selection and budget completion. Equation 4 combines fixed utility scores with marginal coverage that changes as S grows. Utility is standardized over the feasible pool, while coverage gains are standardized over the remaining candidates at each iteration. A zero variance gives zero standardized scores. If $K > |\mathcal{F}_\gamma|$, all feasible candidates are retained and the remaining slots are filled by the highest-utility unselected trajectories. An empty feasible pool therefore yields utility-only selection. Once the budget includes the entire feasible pool, the selected set is independent of λ.

B.4 Computational Complexity

Given $\mathbf{X} \in \mathbb{R}^{M\times N}$, decomposition and cone filtering take $O(NM)$ time. Let $F := |\mathcal{F}_\gamma|$ and $K_{\mathcal{F}} := \min(K, F)$. Evaluating dense quadratic forms across $K_{\mathcal{F}}$ greedy iterations costs $O(K_{\mathcal{F}}FM^2)$, with $O(M^2)$ per rank-one inverse update. Utility sorting for budget completion costs $O(N\log N)$, and memory usage is $O(NM + M^2)$. Computing the complete feasible ordering once for reuse across budgets takes $O(F^2M^2)$ time. These costs cover subset selection after influence estimation.

C Additional Experimental Details

C.1 Tasks and Datasets

We adopt the single-simulator protocol of CUPID (Agia et al., 2025) on the low-dimensional Lift, Square, and Transport tasks from RoboMimic (Mandlekar et al., 2022). We follow its data splits,

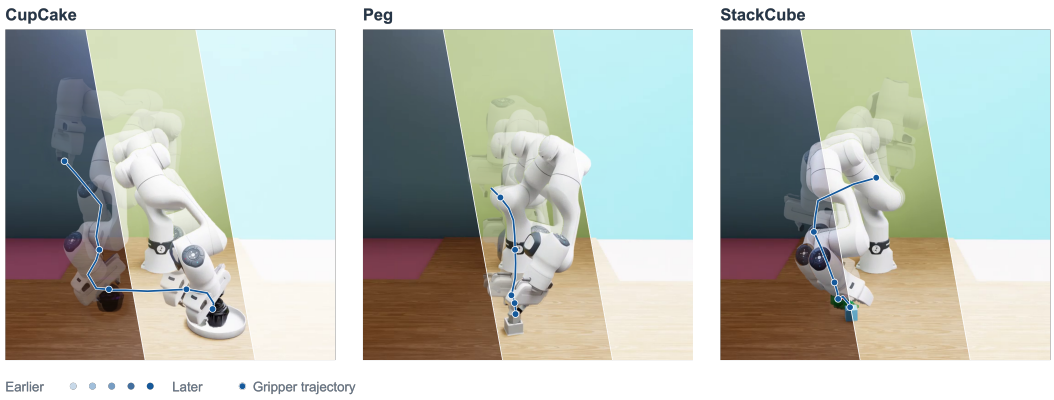

Figure 5: Example simulated executions of `CupCake`, `Peg`, and `StackCube`. Overlaid frames show successive stages of each task, with blue paths tracing gripper motion.

validation protocol, curation budgets, and task-specific training schedules. Filtering is evaluated on all three tasks, while selection is evaluated on Square and Transport.

The Sim-to-Sim and Sim-to-Real study uses `Peg`, `StackCube`, and `CupCake` from OmniReset (Yin et al., 2026). For Sim-to-Sim, MuJoCo serves as the target domain and IsaacSim supplies the source demonstrations. We retain the complete target set and curate source trajectories at the budgets defined in Section 5. The resulting subset is combined with the target demonstrations, and policy performance is evaluated in the target environment. Figure 5 illustrates the simulated task executions.

For each task of the Sim-to-Real study, we select 600 demonstrations from a source pool of 1,200 simulation demonstrations. The selected source demonstrations are then combined with real-world demonstrations for co-training. During mixed-domain training, real and simulation samples account for 20% and 80% of the sampling distribution, respectively. Real-world training samples are aligned with individual policy decisions by pairing the observations used for a decision with the action commands actually issued. We pad these samples to the policy prediction horizon and exclude padded action positions from the training loss.

C.2 Simulation Policy Training Details

The single-simulator experiments use the convolutional Diffusion Policy configuration released with CUPID (Agia et al., 2025). The policy has a prediction horizon of 16 with two observation steps and eight action steps. Each curated subset is trained from scratch. Methods compared at the same task and budget share the policy configuration, training schedule, and normalization procedure.

The Sim-to-Sim experiments use a state-based behavior-cloning policy that takes two consecutive 200-dimensional observations. The resulting 400-dimensional input is processed by an MLP with widths $400 \to 1024 \to 512 \to 512 \to 512 \to 7$ and ReLU activations. We minimize mean-squared action error for 50,000 gradient steps using AdamW. The batch size is 256 and the learning rate is 10^{-4}. AdamW uses betas $(0.95, 0.999)$ and weight decay 10^{-6}. The learning rate is warmed up for 500 steps and then follows cosine decay. Gradients are clipped to unit norm. Evaluation uses exponential moving average weights with power 0.75 and maximum decay 0.9999. Every policy is trained from scratch on all target demonstrations and the selected source trajectories. Training examples are sampled uniformly from the combined data, while the normalization statistics are computed from the target set.

C.3 Real-World Policy Training Details

Base policy and scoring rollouts. We obtain the base-policy weights by training a task-specific image-conditioned Diffusion Policy on simulated demonstrations using behavior cloning with a diffusion denoising objective. We deploy this simulation-pretrained policy on the physical robot and

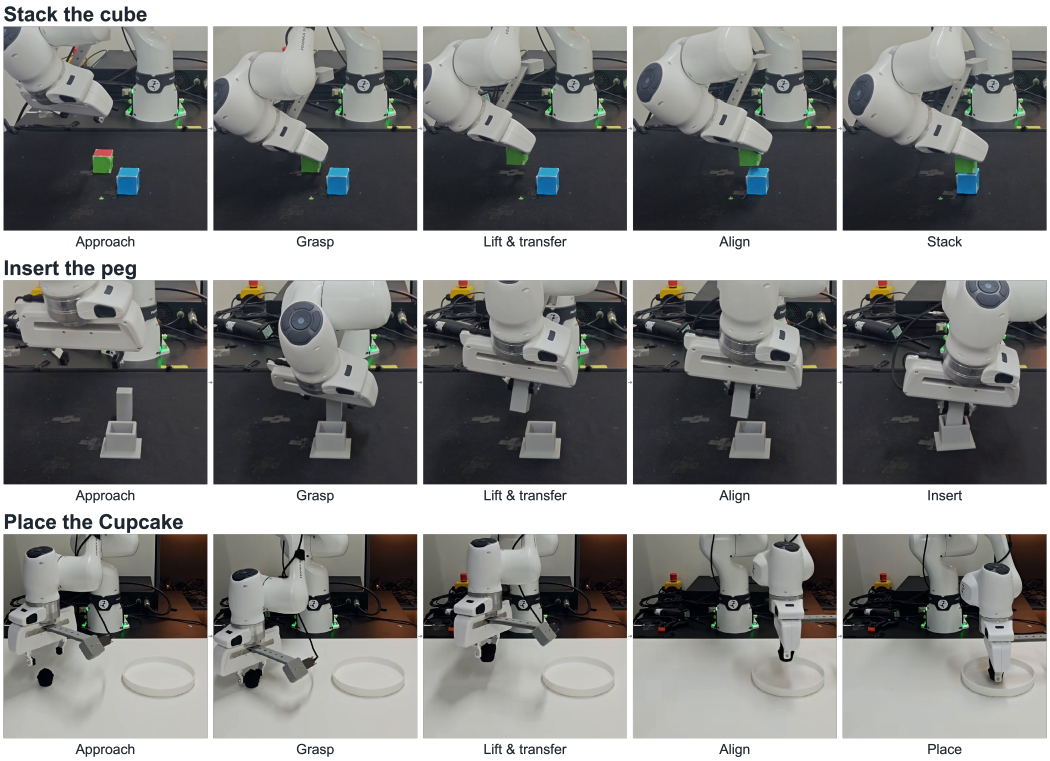

Figure 6: Example real-world executions of `StackCube`, `Peg`, and `CupCake`, shown from top to bottom. Frames progress from left to right through grasping, transfer, and final placement or insertion.

retain successful executions confirmed by the operator. These recorded trajectories supply both the target demonstrations for co-training and the target-domain scoring rollouts for curation. The base-policy weights remain frozen when computing source and target attribution gradients. After selection, we initialize a new policy and train it on all target demonstrations together with the curated source subset. Final evaluation uses separate real-world executions.

Robot platform and cameras. We use a Franka Research 3 robot equipped with a Franka Gripper. The camera setup consists of an Intel RealSense D405 mounted at the wrist, an Intel RealSense D435 providing the side view, and an Intel RealSense L515 providing the head view. The policy receives RGB observations from all three cameras together with proprioceptive observations consisting of the seven arm joint positions and the gripper width.

Task objects and environment alignment. The task objects are fabricated using a Bambu Lab P1S 3D printer. We use corresponding objects in simulation and the physical workspace, calibrate the cameras, and match the object-placement regions across domains. This alignment reduces differences arising from object geometry, camera placement, and workspace configuration while retaining the dynamics and visual differences encountered during physical execution. Figure 6 shows the corresponding task executions on the physical robot.

Policy architecture. We use an image-conditioned Diffusion Policy with a separate ResNet-18 encoder for each camera. The visual features are concatenated with the proprioceptive observations from two consecutive time steps and supplied as global conditioning to a temporal U-Net. Input images are resized to 84×84 pixels and randomly cropped to 76×76 pixels during training. The U-Net predicts a sequence of eight-dimensional actions, with seven arm joint commands and one gripper command per time step. We use a prediction horizon of 16 and a DDPM noise scheduler with 100 diffusion steps. The image encoders and action network are trained jointly.

Table 3: Policy and training configuration for sim-to-real co-training.

Parameter	Value
Observation history	2 steps
Prediction horizon	16 steps
Image resolution	84×84
Training crop	76×76
Visual encoder	ResNet-18 per camera
U-Net channel widths	256, 512, 1024
Diffusion steps	100
Noise schedule	Cosine
Optimizer	AdamW
Learning rate	10^{-4}
AdamW betas	$(0.95, 0.999)$
Weight decay	10^{-6}
Learning-rate schedule	Cosine decay
Warmup	500 optimization steps
Batch size	64
Training epochs	150
Real / simulation sampling	20% / 80%
Maximum EMA decay	0.9999

Optimization. We train the policies from scratch for 150 epochs using AdamW with a batch size of 64. The learning rate is warmed up for 500 optimization steps and then follows a cosine decay schedule. We minimize the mean-squared error between the predicted and sampled diffusion noise over valid action positions. We maintain an exponential moving average of the policy parameters with a maximum decay of 0.9999. The training configuration is summarized in Table 3.

C.4 Baseline Implementations

All baselines curate complete trajectories from the same candidate pool. Filtering removes the lowest-ranked demonstrations from the initial training split. Selection adds the highest-ranked source trajectories to the fixed target set. Random samples trajectories without replacement. Target-only and full-pool controls use the same policy and optimization protocol as the curated subsets. Scalar-score methods share a fixed trajectory ordering across budgets. Budget-dependent methods receive the requested cardinality as part of their subset objective.

In Sim-to-Real, policy-conditioned scoring uses the shared simulation-pretrained base policy, and target-conditioned methods receive the same recorded successful real-world data. The selected source subsets are combined with this fixed target set for training from scratch.

DemInf. We use the released DemInf implementation for single-simulator filtering (Hejna et al., 2025b). For Sim-to-Sim, the identity encoder is applied to the state and action. The two feature groups are standardized separately over the source pool. We estimate their local dependence with the KSG estimator using Chebyshev distance and $k = 5$. Frame-level estimates are averaged within each trajectory, and trajectories with larger dependence are selected.

Demo-SCORE. The single-simulator implementation follows the adaptation in CUPID (Chen et al., 2025; Agia et al., 2025). For Sim-to-Sim, each target-rollout frame inherits its rollout success label. Actions are standardized using the target-rollout statistics. We train a $7 \to 8 \to 8 \to 1$ classifier for 200 epochs using dropout 0.3 and weighted binary cross entropy. The optimizer is AdamW with learning rate 10^{-4} and weight decay 0.1. A source trajectory is scored by the mean predicted success probability of its actions.

Success Similarity. The single-simulator implementation follows CUPID and uses low-dimensional state observations (Agia et al., 2025). For Sim-to-Sim, source states and target-rollout states are jointly min-max normalized. We compute the mean pairwise Euclidean distance between

every source trajectory and each successful target rollout. The negative distance averaged over successful rollouts defines the trajectory score. For Sim-to-Real, we encode image and proprioceptive observations with the frozen Diffusion Policy observation encoder and average the features over the sampled windows of each candidate or real rollout. We rank candidates by their mean cosine similarity to the successful real-rollout representations.

CUPID. We follow the closed-loop attribution procedure released with CUPID (Agia et al., 2025). The single-simulator implementation evaluates each transition over 64 diffusion-timestep and noise samples before applying a 4,000-dimensional TRAK projection. For Sim-to-Sim, we compute full-policy gradients of the behavior-cloning loss and use the same projection dimension. In these simulated settings, frame-level influences are summed within source demonstrations and target rollouts. The resulting utility adds influence on successful rollouts and subtracts influence on failed rollouts. For Sim-to-Real, we average projected gradients within each source candidate and real rollout, then rank candidates by their mean estimated influence across the recorded successful rollouts. All real scoring rollouts receive positive return labels.

Quality over Quantity. We preserve the normalized policy-gradient similarity of Quality over Quantity (Lee et al., 2026). Diffusion Policy gradients are computed from the final denoising block and compressed to 2,048 dimensions using OPORP. We sample at most 32 training windows from each demonstration. Each window receives its maximum similarity to a validation window, followed by averaging within its trajectory. For Sim-to-Sim, we use the normalized Gaussian log-likelihood gradient of the action-head weight and bias. The gradients are compressed to 1,024 dimensions. Each source frame is matched to its most similar target frame from a reference set capped at 4,096 frames. These similarities are then averaged within the source trajectory.

PSD. We implement the spectral quality metric of Sojib et al. (2026) using the end-effector Cartesian position. A full discrete Fourier transform is applied independently to each coordinate without detrending or removing its constant component. Spectral power is summed across coordinates and frequencies. Trajectories with lower total power receive higher priority.

DataMIL. For single-simulator Diffusion Policy, we use a one-step adaptation of the DataMIL metagradient estimator on the final denoising block (Dass et al., 2026). Each source trajectory receives a differentiable weight, and its score is obtained by differentiating target validation loss through the inner update. For Sim-to-Sim, all features use target-domain normalization and every fourth frame is retained for attribution. We reserve 100 target trajectories for meta-validation. The pretrained policy trunk is frozen, while the action head is differentiably unrolled for 200 updates with learning rate 10^{-3} and batch size 256. Each update contains equal numbers of target and weighted source examples. The negative derivative of target validation error with respect to each trajectory weight gives its score. All target trajectories are restored for final co-training. For Sim-to-Real, we use a one-step gradient-alignment adaptation. We compute full-policy gradients of the squared denoiser-output objective and project them to 2,048 dimensions, averaging over sampled windows within each source candidate and real rollout. The reference gradient is the mean across successful real rollouts. We rank source candidates by their gradient inner product with this reference and retain all target demonstrations for final co-training.

FAKTUAL. For simulated experiments, we use the exact level-two trajectory signature and normalized signature Gram matrix of FAKTUAL (Sirigiri et al., 2026). In the single-simulator setting, each path contains the object state, end-effector pose, gripper state, and raw action. All path features are standardized and augmented with normalized time, then a zero base point is prepended. We follow the entropy and determinant selection procedure with entropy fraction 0.5. Its stochastic-greedy tolerance is 0.1 and determinant regularization is 10^{-8}. For Sim-to-Sim, the state-action path retains the most recent value of each history-based state term. We use determinant-greedy selection with the regularized objective $\log\det(\mathbf{K} + 10^{-4}\mathbf{I})$. The selected prefixes provide the subsets at different budgets. For Sim-to-Real, we adapt the determinant-based selector to the projected full-policy gradient representations used by the DataMIL adaptation above. We normalize these source representations to construct their Gram matrix and greedily select candidates by the log-determinant gain with regularization 10^{-4}. This adaptation measures diversity in policy-gradient space and uses only source representations for subset selection.

SIEVE. We preserve the segmentation, primitive discovery, and pattern-aware selection stages of SIEVE (Wu et al., 2026). A gripper-command transition defines a segment boundary after the new state persists for five frames. Each segment is represented by standardized state-action features at its start, middle, and end. The single-simulator implementation uses the complete low-dimensional observation, while Sim-to-Sim uses the most recent value of each state term and the current action. The descriptors are projected to at most 256 principal components. We search 20 candidate primitive counts on a 10% trajectory subset, using minimum average cluster sizes of 20 and 1,000 for the two settings. Each trajectory is represented as an ordered primitive pattern. Budgets are allocated by the marginal coverage of primitives and transitions, while cosine-medoid distance orders demonstrations that share a pattern.

TAROT. We adapt the gradient transport representation of TAROT (Feng et al., 2025). Diffusion Policy trajectories use the mean DDPM-loss gradient of the final denoising block. Sim-to-Sim trajectories use the mean action-head gradient over the trajectory. Both representations are compressed to 2,048 dimensions and jointly centered. They are whitened with covariance regularization 10^{-5} and normalized to unit length. Selection expands source neighborhoods around target trajectories. An entropic optimal-transport solve with regularization 0.01 and 200 iterations resolves the final neighborhood when it exceeds the remaining budget. We disable data-repetition weights so that each method changes only the selected source subset.

C.5 Evaluation Protocol

Final evaluation rollouts are collected from the retrained policies and are separate from the scoring rollouts used for curation. In the simulated OmniReset tasks, `Peg` succeeds when a 160-step rollout reaches position error below 0.01 m and roll-plus-pitch error below 0.1 rad. `StackCube` requires stable pose and object contact after release and settling, with position error below 0.005 m and roll-plus-pitch error below 0.025 rad. `CupCake` requires the same pose thresholds to hold for five consecutive steps within a 160-step rollout.

For each real-world task and method, we evaluate the checkpoint from the final training epoch over 40 trials. The lighting-shift experiment evaluates the same checkpoint over 20 additional trials.

C.6 Influence-Space Visualization

To examine how different curation criteria balance utility and coverage, we visualize the `StackCube` subsets selected at a 10% source budget. Figure 7 projects the orthogonal influence components onto two principal components and colors selected demonstrations by utility percentile. CUPID concentrates high-utility trajectories in a narrow region, indicating repeated closed-loop influence patterns. Diversity-based methods cover a broader region but include more low-utility trajectories because their selection spaces are not grounded in target-policy performance. TUCO retains high-utility candidates while spanning a broader range of influence directions, illustrating how closed-loop utility and marginal coverage jointly shape the selected subset.

D Extended Experiments

D.1 Hyperparameter Sensitivity

We study how the utility–coverage weight λ and the cone threshold γ affect subset selection. To isolate their effects, we vary λ with $\gamma = 0.01$ and vary γ with $\lambda = 0.2$.

Effect of λ. Table 4 presents the complete single-simulator results. Averaged across all 25 task and budget configurations, $\lambda \in \{0.1, 0.2, 0.5\}$ achieves success rates of 59.76%, 61.10%, and 60.18%, respectively. The configuration $\lambda = 0.2$ obtains the highest or tied-highest result in 17 configurations and performs consistently in both filtering and selection. The best value still varies in individual cases. Square selection at the 75% budget changes by up to 9.8% across the tested values, while Transport selection at the 10% budget favors $\lambda = 0.5$. The preferred balance between utility and coverage can therefore depend on the task and subset size.

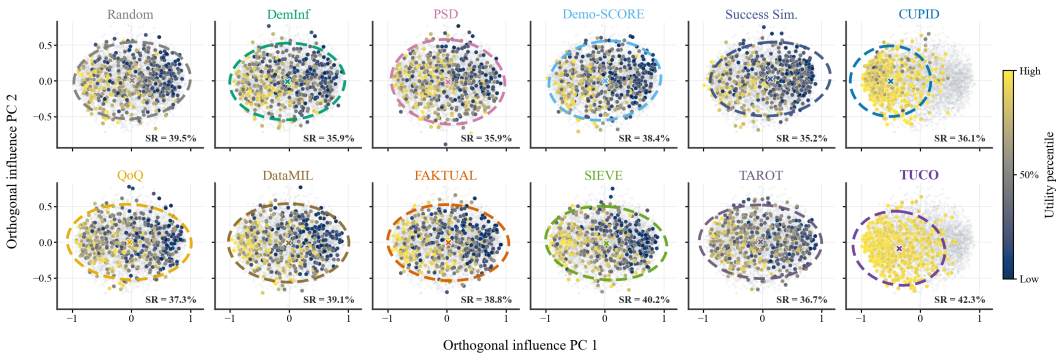

Figure 7: Influence-space visualization of `StackCube` subsets selected at a 10% source budget. Gray points denote the source pool and colored points denote selected demonstrations by utility percentile. Ellipses summarize the PCA-projected spread of each subset.

Table 4: Sensitivity to λ in the single-simulator setting with $\gamma = 0.01$. Entries are target-domain success rates reported as mean $\pm$ standard deviation. Bold parameter values denote the default configuration.

Protocol	Task	λ	10%	25%	50%	75%	90%
Filtering	Lift	0.1	1.000±0.000	1.000±0.000	1.000±0.000	0.986±0.013	0.960±0.023
		0.2	1.000±0.000	1.000±0.000	1.000±0.000	1.000±0.000	0.988±0.016
		0.5	1.000±0.000	1.000±0.000	1.000±0.000	0.988±0.009	1.000±0.000
Filtering	Square	0.1	0.760±0.006	0.710±0.024	0.730±0.013	0.534±0.029	0.364±0.026
		0.2	0.762±0.005	0.732±0.018	0.734±0.014	0.528±0.021	0.360±0.024
		0.5	0.738±0.011	0.700±0.019	0.706±0.020	0.536±0.031	0.388±0.021
Filtering	Transport	0.1	0.400±0.012	0.370±0.019	0.372±0.024	0.246±0.015	0.150±0.013
		0.2	0.412±0.013	0.420±0.020	0.334±0.015	0.268±0.009	0.196±0.017
		0.5	0.398±0.010	0.400±0.014	0.402±0.031	0.268±0.017	0.148±0.015
Selection	Square	0.1	0.592±0.019	0.594±0.012	0.628±0.016	0.680±0.010	0.714±0.013
		0.2	0.572±0.034	0.600±0.009	0.634±0.015	0.764±0.016	0.758±0.010
		0.5	0.572±0.026	0.592±0.011	0.670±0.013	0.666±0.014	0.738±0.012
Selection	Transport	0.1	0.382±0.013	0.418±0.018	0.488±0.027	0.462±0.012	0.400±0.010
		0.2	0.344±0.011	0.452±0.016	0.484±0.021	0.476±0.015	0.458±0.009
		0.5	0.424±0.009	0.396±0.013	0.412±0.022	0.475±0.016	0.428±0.012

Table 5 extends this analysis to Sim-to-Sim co-training. Performance becomes more sensitive to λ under simulator shift, especially at low and intermediate budgets. On `StackCube`, the success rate at the 10% budget ranges from 32.9% to 42.3%. On `CupCake`, $\lambda = 0.5$ performs best at the 25% budget. Across all 18 configurations, $\lambda = 0.2$ obtains the highest or tied-highest result in 15 cases and the highest budget-averaged success rate on every task. These results reveal task-dependent sensitivity while supporting $\lambda = 0.2$ as a stable shared configuration for the single-simulator and Sim-to-Sim settings.

Effect of γ. We next study the cone threshold in Sim-to-Sim co-training with $\lambda = 0.2$. As shown in Table 6, $\gamma = 0.01$ outperforms $\gamma = 0.1$ in 12 of the 18 task and budget configurations and matches it in five more. The larger threshold produces a higher result only on `CupCake` at the 10% budget, where the difference is 0.5%. Its lower performance on `Peg` and `StackCube` suggests that a narrow feasible cone can exclude trajectories with useful complementary effects. The setting $\gamma = 0.01$ retains a broader pool of performance-aligned candidates and yields more consistent performance across tasks.

Table 5: Sensitivity to λ in Sim-to-Sim co-training with $\gamma = 0.01$. Entries are target-domain success rates reported as mean ± standard deviation. Bold parameter values denote the default configuration.

Task	λ	5%	10%	15%	25%	50%	75%
Peg	0.1	0.572±0.044	0.588±0.045	0.634±0.052	0.678±0.010	0.709±0.040	0.674±0.013
	0.2	0.567±0.033	0.614±0.026	0.633±0.031	0.716±0.023	0.709±0.040	0.674±0.013
	0.5	0.564±0.069	0.591±0.030	0.632±0.045	0.704±0.051	0.709±0.040	0.674±0.013
StackCube	0.1	0.373±0.036	0.329±0.054	0.387±0.031	0.369±0.031	0.429±0.038	0.426±0.029
	0.2	0.399±0.040	0.423±0.044	0.432±0.028	0.421±0.021	0.455±0.015	0.426±0.029
	0.5	0.387±0.035	0.387±0.050	0.375±0.029	0.376±0.010	0.429±0.038	0.426±0.029
CupCake	0.1	0.503±0.025	0.513±0.029	0.527±0.013	0.536±0.033	0.522±0.040	0.495±0.035
	0.2	0.525±0.033	0.522±0.045	0.541±0.026	0.545±0.033	0.522±0.040	0.495±0.035
	0.5	0.505±0.025	0.505±0.033	0.523±0.039	0.562±0.031	0.522±0.040	0.495±0.035

Table 6: Sensitivity to γ in Sim-to-Sim co-training with $\lambda = 0.2$. Entries are target-domain success rates reported as mean ± standard deviation. Bold parameter values denote the default configuration.

Task	γ	5%	10%	15%	25%	50%	75%
Peg	**0.01**	0.567±0.033	0.614±0.026	0.633±0.031	0.716±0.023	0.709±0.040	0.674±0.013
	0.1	0.559±0.007	0.596±0.044	0.610±0.054	0.659±0.036	0.709±0.040	0.674±0.013
StackCube	**0.01**	0.399±0.040	0.423±0.044	0.432±0.028	0.421±0.021	0.455±0.015	0.426±0.029
	0.1	0.364±0.026	0.360±0.021	0.421±0.039	0.404±0.024	0.440±0.015	0.426±0.029
CupCake	**0.01**	0.525±0.033	0.522±0.045	0.541±0.026	0.545±0.033	0.522±0.040	0.495±0.035
	0.1	0.475±0.035	0.527±0.031	0.523±0.034	0.540±0.052	0.522±0.040	0.495±0.035

Sensitivity decreases at larger selection budgets, where several parameter values produce identical results. As the budget grows, the selected subset covers the feasible pool and the remaining positions follow the same utility ranking. Different configurations consequently converge to the same subset. The effects of λ and γ are more visible at lower budgets, where each selection must resolve a stronger trade-off among candidate trajectories.

D.2 Robustness to Unseen Lighting Conditions.

We examine whether TUCO retains its advantage under an unseen visual shift by introducing an RGB light strip around the peg and hole during Peg evaluation (Figure 8). We evaluate each policy over 20 trials. Table 7 shows that TUCO achieves the highest success rate at 25.0%, exceeding both CUPID and target-only training by 10 percentage points, while full-pool co-training fails every trial. Despite the substantial performance drop under altered lighting, TUCO maintains its advantage over the evaluated baselines.

E Extended Discussions

E.1 Insights into Robust Curation across Domain Gaps

Our study shows that the same TUCO criterion remains effective in single-simulator, Sim-to-Sim, and Sim-to-Real settings. This result supports grounding curation in observed target-domain behavior as the domain gap changes. Source-domain quality, latent similarity, and trajectory geometry can provide useful proxies, but they may increasingly reflect rendering, sensing, or dynamics differences across domains. TUCO represents each source demonstration through its estimated effects on shared target-domain scoring rollouts. In Sim-to-Real, utility measures aggregate support for recorded successful executions, while coverage measures complementary influence patterns across them. This feedback supports the same decomposition and selection mechanism using a simulation-pretrained base policy.



Figure 8: Physical `Peg` setup with the RGB light strip enabled at test time. The strip surrounds the peg and hole and changes illumination and color within the workspace without moving any object.

Table 7: `Peg` success rates (%) with the RGB light strip enabled at test time. The light-strip column uses 20 trials per method. The standard-lighting column repeats the 40-trial entries of Table 1.

Method	Standard lighting	RGB light strip
10 real demos	15.0	15.0
10 real demos + all sim data	17.5	0.0
CUPID	75.0	15.0
TUCO (ours)	**82.5**	**25.0**

Our study also reveals that performance grounding alone is insufficient when the output is a dataset. The value of a source trajectory depends on the fixed target demonstrations and the influence modes already represented by the selected subset. Independent ranking can repeatedly select trajectories with similar training effects. TUCO addresses this dependency through set-level marginal coverage while retaining direct target-performance guidance. The consistency observed across domain gaps therefore supports two general design principles for curation: evaluate data through downstream target behavior and optimize the composition of the selected set.

E.2 Source Composition as a Co-Training Design Variable

To the best of our knowledge, this study provides the first systematic evidence that active curation improves both Sim-to-Sim and Sim-to-Real co-training. More broadly, it introduces a new perspective on co-training: in addition to choosing the source-to-target mixing ratio, one should decide which source demonstrations occupy the source portion of that mixture. The source-to-target demonstration ratio specifies the relative amounts of available training data, while curation determines which source behaviors are included. Our comparisons with the full source pool establish the benefit of selecting a subset. Comparisons with random selection at the same source budget and with fixed target demonstrations further isolate the value of source-data composition.

This distinction is particularly important for large simulation pools. Simulation can cover behaviors missing from scarce target data, but it can also overrepresent redundant trajectories and domain-specific strategies. Active curation treats this pool as a candidate library and retains demonstrations that support target success while complementing existing target data. Our findings therefore shift the co-training question from only *how much simulation data should be mixed* toward *which simulation demonstrations should be mixed*. This additional design dimension turns simulation scale into more effective cross-domain supervision and creates a path toward stronger policies with lower training and real-data collection costs.

E.3 Limitations

Automatic curation budget. TUCO optimizes which source demonstrations to retain under a prescribed curation budget and source-to-target mixing ratio. This design isolates the effect of source-data composition and supports controlled comparison across curation methods. In deployment, however, the appropriate subset size is typically unknown and may vary with the domain gap and the behavioral support already provided by the target demonstrations. A small budget may omit useful source behaviors, while a large budget may reintroduce redundancy or source-specific patterns after the most valuable candidates have been selected. TUCO evaluates set-dependent marginal value under a fixed cardinality constraint and therefore provides no calibrated boundary between demonstrations that should be retained and those that provide no further benefit. This limitation reflects a common formulation in data curation, where candidates receive selection scores and the final subset is determined by a predefined budget. We believe that it is an important direction to automatically

identify the useful portion of a source pool, independent of a predefined retention ratio. We leave joint optimization of subset composition and cardinality for future work.

Broader transfer settings. Our evaluation covers manipulation tasks across single-simulator, Sim-to-Sim, and Sim-to-Real settings with compatible observation and action interfaces. This scope allows source and target demonstrations to train the same policy and makes their influence profiles directly comparable. Cross-task or cross-embodiment transfer may additionally require aligning observations, actions, and task semantics before source demonstrations can be evaluated in a shared closed-loop space. Large vision-language-action policies may also produce different influence geometry from the task-specific policies studied here. We also leave adaptation and evaluation of TUCO under these broader transfer settings for future work.