Using Non-Expert Data to Robustify Imitation Learning via Offline Reinforcement Learning

Kevin Huang*[1], Rosario Scalise*[1], Cleah Winston[1], Ayush Agrawal[1], Yunchu Zhang[1], Rohan Baijal[1], Markus Grotz[1], Byron Boots[1], Benjamin Burchfiel[2], Hongkai Dai[2], Masha Itkina[2], Paarth Shah[2], and Abhishek Gupta[1]

***Abstract*— Imitation learning has proven effective for training robots to perform complex tasks from expert human demonstrations. However, it remains limited by its reliance on high-quality, task-specific data, restricting adaptability to the diverse range of real-world object configurations and scenarios. In contrast, non-expert data—such as play data, suboptimal demonstrations, partial task completions, or rollouts from suboptimal policies—can offer broader coverage and lower collection costs. However, conventional imitation learning approaches fail to utilize this data effectively. To address these challenges, we posit that with right design decisions, offline reinforcement learning can be used as a tool to harness non-expert data to enhance the performance of imitation learning policies. We show that while standard offline RL approaches can be ineffective at actually leveraging non-expert data under the sparse data coverage settings typically encountered in the real world, simple algorithmic modifications can allow for the utilization of this data, without significant additional assumptions. Our approach shows that broadening the support of the policy distribution can allow imitation algorithms augmented by offline RL to solve tasks robustly, showing considerably enhanced recovery and generalization behavior. In manipulation tasks, these innovations significantly increase the range of initial conditions where learned policies are successful when non-expert data is incorporated. Moreover, we show that these methods are able to leverage *all* collected data, including partial or suboptimal demonstrations, to bolster task-directed policy performance. This underscores the importance of algorithmic techniques for using non-expert data for robust policy learning in robotics.**

I. Introduction

Imitation learning has enabled remarkable progress in robot learning, training reactive closed-loop policies from high-quality demonstrations via supervised learning using expressive policy classes such as diffusion models parameterized with large neural networks [1], [2], [3]. Despite impressive performance under conditions similar to the training distribution, these policies can be quite brittle beyond this setting. They show vulnerability to out-of-distribution (OOD) scenarios, where even minor deviations in object configurations or environmental conditions can lead to failure [4]. A natural way to address policy fragility is to simply collect more expert data, broadening the coverage of expert demonstrations. The challenge is that collecting such data can be expensive and scale poorly, requiring an impractical amount of data collection to cover combinatorial scene conditions. As a result, policies trained with standard imitation learning can struggle to handle real-world variability on deployment.

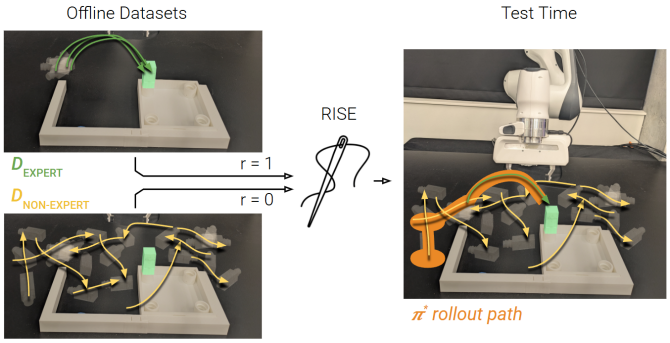


Fig. 1: RISE allows non-expert data to be used alongside expert data to provide robust, high-coverage behavior for robotic manipulation. This allows for *all collected data* to be used to imbue the policy with generalization and recovery behavior.

The formulation of imitation learning through supervised learning requires (near) optimal task demonstrations, which have to be carefully curated per task and can be difficult to collect at scale, especially for high precision or high dexterity problems. In most large-scale data collection efforts, not all data satisfies these criteria. This results in a significant fraction of collected data being discarded through the process of data curation/filtering [5], [6], [7], despite this data containing useful information about the dynamics of the world. This begs the question - *can cheaper sources of non-optimal data beyond expert, task-specific demonstrations be used to improve the performance of imitation learning?* In this work, we study how cheaper and typically abundant data sources such as undirected play data, unsuccessful/partial demonstrations (whether from a human demonstrator or policy rollouts), or data from other tasks can be made useful for improving the robustness of imitation learning.

We study this problem through the lens of offline reinforcement learning (RL). Typical offline RL algorithms [8], [9], [10] treat all data (whether it be optimal or suboptimal) as arbitrary off-policy data and synthesize optimal policies through reward-based, temporal-difference learning [11]. Directly applying standard offline RL methods for leveraging suboptimal data can become challenging for real-world problems, since reward can be challenging to specify without considerable domain knowledge or privileged information. In this work, we propose a simple alternative, instantiating an offline RL algorithm that can learn from simple binary

*Equal contribution. Correspondence to {kehuang, rosario}@cs.washington.edu
[1] University of Washington
[2] Toyota Research Institute (TRI)
Paper website: https://uwrobotlearning.github.io/RISE-offline/

rewards, 1 for optimal data and 0 for suboptimal data. This allows the learning algorithm to leverage suboptimal data to learn how to *recover* the system back to states in the expert state distribution, while replicating the expert behavior on these expert-provided states. This naturally robustifies the policy to solve tasks from a diversity of states beyond the narrow range of states provided by the expert, while requiring minimal infrastructural modifications and assumptions beyond that of typical imitation learning.

While offline RL via dynamic programming can in principle "stitch" together useful segments from suboptimal data for data-efficient recovery, we find that practical instantiations of offline RL methods in high-dimensional state-action spaces can fail to demonstrate this stitching capability without an impractically high degree of data coverage. To address the challenges resulting from the lack of data coverage, we introduce a notion of "fuzziness" into the state representation for the policy network by enforcing a notion of local smoothness via Lipschitz continuity. For recoverable OOD states, doing so significantly improves the policy's ability to "stitch" offline data. This enables suboptimal data to easily be used for improving the robustness of imitation learning, even in low data coverage regimes.

We make the following contributions - 1) we introduce an offline RL framework for leveraging non-expert data to robustify imitation learning policies, 2) we show the pitfalls of standard offline RL in the low data regime, and introduce the Robust Imitation by Stitching from Experts (`RISE`) algorithm, to improve trajectory stitching 3) We show that `RISE` is effective across various types of non-optimal data - ranging from undirected play data to suboptimal demonstrations or policy evaluation rollouts, and even multitask datasets, 4) We demonstrate the efficacy of `RISE` on various tabletop manipulation tasks in simulation and furniture assembly tasks on a real robot.

II. Related Work

Imitation Learning: Imitation learning methods aim to learn closed loop policies from near optimal demonstration data. This is a well studied field, with a plethora of work [12], [13], [14], [15] on methods and applications. Work in imitation learning has primarily focused on either dealing with compounding error [16], [17], [18], [19], [20], incorporating richer generative distributions [1], [21], [22] or using robust policy backbones [23], [24], [25], [26]. In this work, we show that in addition to expert demonstrations, *non-expert* data can be leveraged to robustify imitation learning.

Offline Reinforcement Learning: Offline reinforcement learning is a closely related subarea of research, where pre-collected off-policy datasets are used to synthesize task-directed behavior [27]. These methods do not typically assume that pre-collected data is optimal, instead using rewards to infer which behaviors are optimal from offline datasets. While offline RL methods come in many forms - importance sampling based [28], [29], model-based [30], [31], dynamic-programming-based [32], [9], [8], [10], many of these methods operate on the principle of *pessimism* - assuming the worst outside of the training distribution, restricting the synthesized behavior to compositions of behaviors within the training distribution, often referred to as "stitching". Importantly, most of these methods still rely on access to rewards at training, an often onerous assumption that makes these methods difficult to use. Perhaps most relevant to this work is SQIL [33], which performs offline RL on a mixture of optimal demonstration data and suboptimal data. Our findings indicate that in sparse-coverage problems, SQIL can be insufficient for data stitching, and thus, we propose an alternative that allows for better transitions from non-expert data.

Out-of-distribution Recovery: A set of prior methods have considered techniques for recovery back to the manifold of expert behavior, so as to robustify learned policy behavior [34], [19], [17], [35]. [34] aims to use keypoint driven gradients to recover to the training distribution, using explicit pose and keypoint estimation and an inverse controller. [35] uses equivariance to learn a recovery controller back to the expert manifold. In contrast, `RISE` does not rely on explicit object and state representations, and does not have to learn a separate policy and recovery controller. Prior work does local recovery using synthetic data, via generative models [19] or learned dynamics [17]. Since these models are only valid in local regions around the data, they struggle with global notions of recovery, as is enabled by `RISE`. Perhaps most relevant is [36], which identifies sub-trajectories in suboptimal data that recover to expert states and selectively adds these to imitation learning. We show that `RISE` is significantly more performant and data efficient than [36] due to the ability to stitch trajectories.

III. Background

Imitation Learning: We consider an episodic finite-horizon MDP given by $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, p, r, \gamma, H\}$, with standard notation [11]. A policy π is a function that maps $s \in \mathcal{S}$ to a distribution over $a \in \mathcal{A}$, and its optimality can be measured by $J(\pi) := \mathbb{E}\left[\sum_{t=0}^{H} \gamma^t r(s_t, a_t) | s_0 \sim p_0, a_t \sim \pi(\cdot|s_t)\right]$. In the imitation learning problem, we are given a set of demonstrations $\mathcal{D}_E = \{(s_j, a_j)\}_j$ generated from rolling out a (near) expert policy π_E, often a human demonstrator from an initial state distribution p_0. Given this data, behavior cloning methods learn a policy $\hat{\pi_E}$ via a supervised learning objective: $\hat{\pi_\theta} \leftarrow \max_\theta \mathbb{E}_{(s,a)\sim\mathcal{D}_E}[\log(\pi_\theta(a|s))]$. While we parameterize π as a conditional diffusion model [1], our formulation is equally applicable to π being any expressive generative model [21], [1], [37]. While π_θ is performant for "in-distribution" initial conditions $s_0 \sim p_0(\cdot)$, it can be suboptimal when evaluated from OOD conditions $s_0 \sim p_{\text{test}}(\cdot)$.

Offline Reinforcement Learning: Offline RL learns optimal policies from a fixed offline (potentially suboptimal) dataset of transitions $\mathcal{D} = \{(s, a, s', r)_i\}_{i=1}^N$, without access to online data collection as is common in RL. Offline RL assumes access to labeled rewards r, finding a reward-maximizing policy within the support of the offline data. In offline RL literature [38], a majority adopt the mechanism of off-policy RL with an additional element of "conservatism" to avoid propagating counterfactual OOD value estimates.

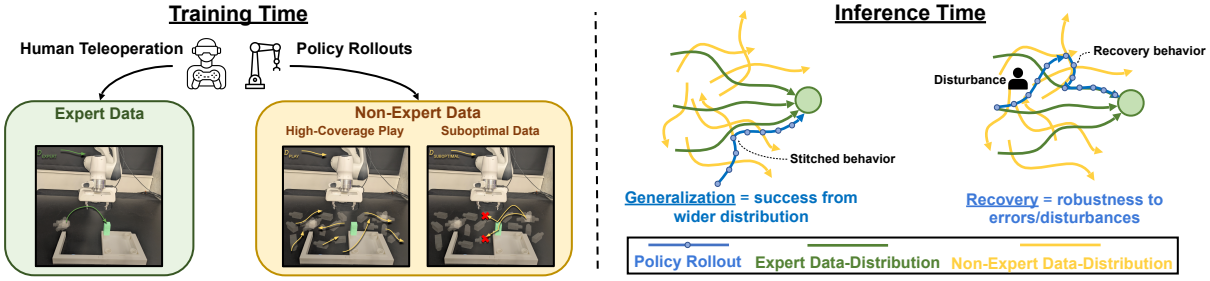


Fig. 2: Various types of data on the `one-leg` task is shown: expert, high-coverage, and suboptimal, which can be collected by a human or from autonomous policy rollouts (for example during evaluation). `RISE` is able to use combinations of different expert and non-expert datasets to improve policy robustness. By stitching trajectories from non-expert data, `RISE` policies can succeed from much wider distributions and are robust to disturbances.

We specifically build on a popular, yet simple offline RL variant – Implicit Diffusion Q-Learning (IDQL) [10], that avoids explicitly imposing conservatism by constraining the policy [9] or regularizing the critic [8]. Instead, this work proposes to be conservative by approximating an expectile τ within the distribution of actions, thereby implicitly imposing the principle of conservatism. IDQL first learns a parameterized Q-function $Q_\phi(s,a)$ and value function V_ψ using the following objective:

$$\mathcal{L}_V(\psi) = \mathbb{E}_{s,a\sim\mathcal{D}}\left[L_2^\tau(Q_\phi(s,a) - V_\psi(s))\right] \quad (1)$$

$$\mathcal{L}_Q(\phi) = \mathbb{E}_{(s,a,s')\sim\mathcal{D}}\left[\left(r(s,a) + \gamma V_\psi(s') - Q_\phi(s,a)\right)^2\right] \quad (2)$$

where $L_2^\tau(x) = |\tau - \mathbb{1}(x<0)|\,x^2$ leads to learning of the τ expectile of the action distribution. Given the Q-value function $Q_\phi(s,a)$, IDQL then extracts the optimal policy $\pi^*(a|s)$ through simple non-parametric test-time optimization – $\pi^*(a|s) = \underset{a\in\{a_1,\ldots,a_K\}\sim\pi_B(a|s)}{\text{argmax}} Q_\phi(s,a)$. Samples are drawn from $\pi_B(a|s)$, the "behavior policy", representing the estimated marginal state-conditional distribution of actions in the training data. $\pi_B(a|s)$ can be obtained through any standard maximum likelihood (or similar) procedure on the offline data, in this case using a diffusion modeling objective [39], [1].

IV. `RISE`: Leveraging Suboptimal Data for Robust Imitation Learning

We begin by describing the problem setting (Section IV-A), followed by an instantiation of a solution technique using offline RL (Section IV-B). We then describe the pitfalls of offline RL methods in low-data regimes, and propose simple algorithmic solutions to these challenges (Section IV-C).

A. Problem Setting: Robustifying Policies with Non-Expert Data

We will assume access to a dataset of expert state-action tuples $\mathcal{D}_\text{E} = \{(s_i,a_i)\}_{i=1}^N$ drawn from an expert, π_E. This is augmented with a dataset of potentially non-expert state-action tuples $\mathcal{D}_\text{NE} = \{(s_i,a_i)\}_{i=1}^M$, where $N \ll M$. The goal is to devise a learning procedure that synthesizes a policy from $\mathcal{D}_\text{E}$ and $\mathcal{D}_\text{NE}$ that maximizes the task performance across a range of initial conditions. Note that the agent is **not** provided with labeled rewards r (as is typical in offline RL) during training, only receiving labels of whether the offline data belongs to the expert dataset $\mathcal{D}_\text{E}$ or the non-expert dataset $\mathcal{D}_\text{NE}$. While non-expert data $\mathcal{D}_\text{NE}$ can take many forms, of particular interest are high-coverage datasets, such as undirected "play" data, multi-task data or closed-loop rollout data collected during evaluations. Partial demonstrations or failures can also provide information about the dynamics of the environment despite being unsuitable for direct imitation. We aim to instantiate a simple, scalable algorithm to augment imitation learning to be able to make use of this non-expert data $\mathcal{D}_\text{NE}$.

B. Learning from Non-Expert Data without Explicit Reward Annotations

While the expert dataset $\mathcal{D}_\text{E}$ can simply be copied via typical behavior cloning [1], it is not as clear how to use $\mathcal{D}_\text{NE}$. We make a simple insight in this work – while non-expert data $\mathcal{D}_\text{NE}$ may not capture expert behavior directly, it conveys information about the dynamics of the environment. This allows a robotic agent to *recover* from an OOD state beyond the expert distribution back to the distribution of expert states in $\mathcal{D}_\text{E}$ (Fig 2). Doing so allows the agent to use the non-expert dataset for robust recovery back to states from which the expert can reliably succeed.

How can we train policies in the absence of an explicitly provided reward function? Drawing inspiration from prior work [33], we can label all (o,a) transitions in the expert dataset with a reward $r=+1$, while labeling all transitions in the non-expert data $\mathcal{D}_\text{NE}$ with reward $r=0$. We can then use these pseudolabeled datasets to learn policies via a typical offline RL procedure, as described in Section III. The resulting updates become:

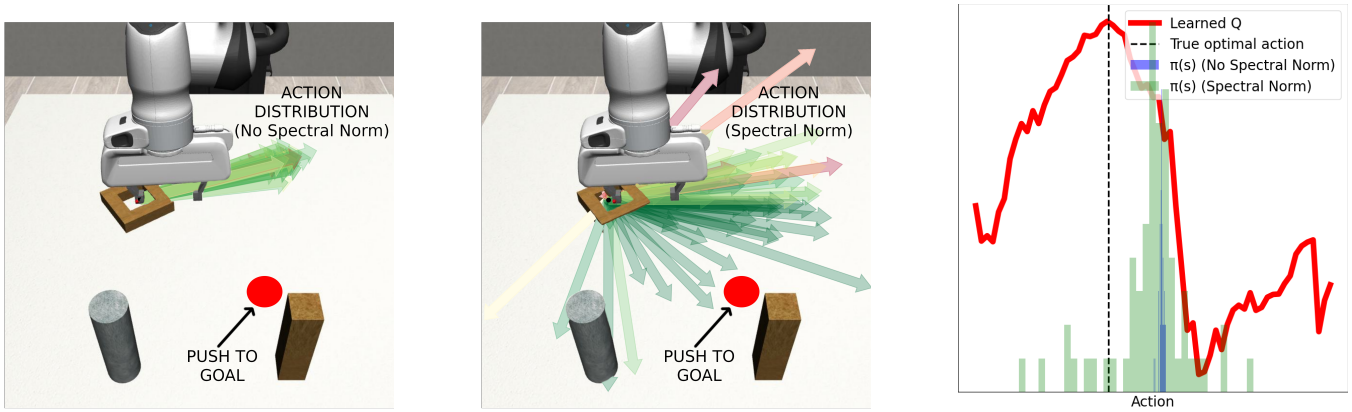
$$\mathcal{L}_V(\psi) = \mathbb{E}_{(s,a)\sim(\mathcal{D}_\text{E}\cup\mathcal{D}_\text{NE})}\left[L_2^\tau(Q_\phi(s,a) - V_\psi(s))\right] \quad (3)$$

$$\mathcal{L}_Q(\phi) = \mathbb{E}_{(s,a)\sim\mathcal{D}_\text{E}}\left[\left(1 + \gamma V_\psi(s') - Q_\phi(s,a)\right)^2\right] + \mathbb{E}_{(s,a)\sim\mathcal{D}_\text{NE}}\left[\left(\gamma V_\psi(s') - Q_\phi(s,a)\right)^2\right] \quad (4)$$

$$\pi_B(a|s) = \text{argmax}_\pi\,\mathbb{E}_{s,a\sim(\mathcal{D}_\text{E}\cup\mathcal{D}_\text{NE})}\left[\log\pi(a|s)\right] \quad (5)$$

$$\pi^*(a|s) = \underset{a\in\{a_1,\ldots,a_K\}\sim\pi_B(a|s)}{\text{argmax}} Q_\phi(s,a). \quad (6)$$

Intuitively, this incentivizes the policy towards state-action transitions in the expert dataset $\mathcal{D}_\text{E}$ while using state-action transitions from the non-expert dataset $\mathcal{D}_\text{NE}$, to provide a path returning to the expert state distribution with no additional cost. Since the update in Equation 3 performs dynamic programming, it can in principle perform data-efficient "stitching" of paths from the non-expert data to recover to expert states. While related in spirit to [33], `RISE` is using $0/1$ rewards for offline RL in continuous RL problems, as opposed to the discrete online RL setting. Naively applying



(a) (b) (c)

Fig. 3: **Visualization of the effect of spectral norm**. (a) On a planar pushing task, using IDQL naively results in an excessively narrow marginal action distribution, leading to poor performance. (b) When a spectral norm penalty is added to the behavior policy loss (Equation (7)), the action distribution is significantly widened. (c) Marginal action distributions projected onto a 1D axis are plotted alongside the learned Q function, which we empirically find to be close to the true Q function in a neighborhood of the data. Narrow action distributions often fail to encompass the optimal action (blue distribution).

this procedure, however, is insufficient in most robotics problems without an impractically high degree of data coverage, as we show empirically (Fig 3). Next, we highlight how to practically improve data "stitchability", allowing policy robustification even in sparse data-coverage problems.

C. Improving Stitchability in Offline Recovery RL

While the methodology in Section IV-B should *in principle* stitch behaviors between non-expert and expert data, or stitch within the non-expert data, we find this is not empirically true across several high-dimensional robotic manipulation problems (Table I). Despite having seemingly high-coverage non-expert data, the likelihood of state-overlap in a continuous space tends to 0, making stitching across exactly overlapping states unlikely. This prevents offline RL from determining paths for recovering to expert states from non-expert ones, even when such paths do exist. While challenging to solve in the most general case, we base our practical improvements on a set of empirical findings in a robotic manipulation setting.

Empirically, we observe that Q-value functions learned with the expectile regression objective [10] tend to be accurate and interpolate well within a neighborhood of the training data, showing reasonable "stitching" behavior. This is visualized by the solid red line in Fig 3 (c) – we can see that despite the state-action coverage being incomplete, the landscape of the Q-function in a neighborhood of the training data is accurate – suggesting that the optimum of the Q-function provides actions that are better than behavior data. However, as shown in prior work, for methods like IDQL, the challenge comes from the *policy extraction* step [40]. While learned Q functions can interpolate in a neighborhood, the marginal action distribution $\pi_B(\cdot|s)$ captured by the behavior policy tends to be overly conservative. The learned distribution overfits to the training set, producing a "narrow" action distribution that fails to encompass optimal actions (see blue policy distribution in Fig. 3(c)). This prevents trajectories from "stitching" together even when they might appear to be close, since sampling-based policy extraction is unable to find the optimal action suggested by the "stitched" value function.

Given this empirical finding, if we assume the learned Q-function is accurate within a neighborhood of actions in the training distribution, we can achieve better performance by explicitly "widening" the marginal base policy distribution π_B. We formalize this notion with the following assumption:

Assumption 1: Let $\mathfrak{N}_d(a|s) := \{a'|d(a,a'|s) < T\}$ denote the neighborhood of an action a at state s, i.e., the actions within T distance under distance metric d. Define $J_\mathcal{D}(\pi) := \mathbb{E}_{a_t\sim\pi(s_t)}\left[\sum_{t=0}^{H}\gamma^t r(s_t,a_t)|s_0\sim\mathcal{D}\right]$. Let $\hat{\pi}(s) = \underset{a\sim\mathfrak{N}(a_0|s),a_0\sim\pi_B(s)}{\text{argmax}} Q_\phi(s,a)$. Then, for any $\delta > 0$, there exists $\mathfrak{N}_d$ such that $|J_\mathcal{D}(\pi) - J_\mathcal{D}(\pi_{opt})| < \delta$, where π_{opt} is the optimal policy.

We find that a natural way to choose such a neighborhood to widen the action distribution of π_B, is to *alias* action distributions between nearby states. In doing so, there is a natural notion of "fuzziness" that is introduced between nearby states, preventing the overly conservative policy behavior mentioned above. We focus on two techniques here:

Enforcing Policy Lipschitz Continuity: One way to implicitly induce aliasing between action distributions at nearby states is to enforce Lipschitz continuity on the policy π_B. This ensures that action distributions at nearby states are similar, avoiding overly conservative action distributions. While there are several ways to enforce Lipschitz continuity, we opt for regularizing the policy with a spectral norm penalty [17], [41]

$$\max_\theta \mathbb{E}_{(s,a)\sim(\mathcal{D}_\text{E}\cup\mathcal{D}_\text{NE})}[\log\pi_\theta(a|s)] + \lambda\sum_{W\in\theta}\|\sigma_{\max}(W)\|^2. \quad (7)$$

Spectral normalization has been shown to bound the Lipschitz constant of a learned model [42]. This objective is simple to optimize using gradient-based supervised learning procedures.

Distance-Based Data Augmentation: An explicit method of widening the distribution of π_B is to augment $\mathcal{D}_U$ with additional transitions in the neighborhood. For a given $(s,a) \in \mathcal{D}_U$, we choose $\mathfrak{N}_d(a|s)$ to be actions from states close to s, as specified by the distance metric d. For every pair of transitions $(s,a),(s',a') \in \mathcal{D}_\text{E}\cup\mathcal{D}_\text{NE}$, we construct an augmented dataset $\mathcal{D}_\text{aug}$ by adding (s,a') to $\mathcal{D}_\text{aug}$ if $d(s,s') < T$ for some distance metric d and threshold T,

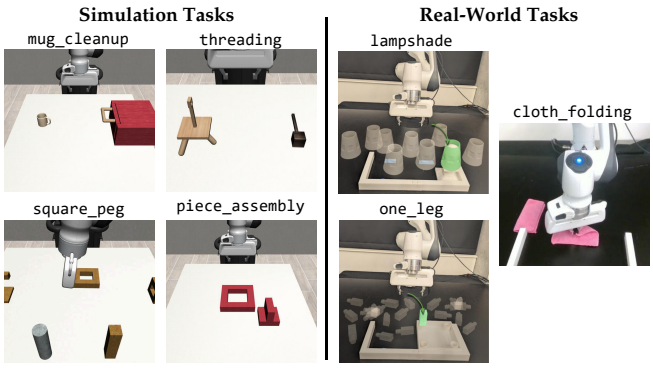


Fig. 4: Depiction of tasks in sim and the real world.

$$\mathcal{D}_{\text{aug}} = \{(s, a') | (s, a), (s', a') \in \mathcal{D}_{\text{E}} \cup \mathcal{D}_{\text{NE}} \text{ if } d(s, s') < T\}. \quad (8)$$

We then train π_B via supervised learning on the entire augmented dataset $\mathcal{D}_{\text{E}} \cup \mathcal{D}_{\text{aug}}$, to learn a broader marginal policy distribution. While the choice of distance metric can vary, we find that using an Euclidean distance in the feature space of a large pretrained vision model, DINOv2 [43], which has been shown to measure meaningful semantic differences between images, is effective. The version of RISE in our experimental evaluation has both spectral norm penalty and distance-based data augmentation included. As we show experimentally, these additions make a significant difference in the ability of RISE to use non-expert data for robust policy learning. In summary, RISE provides a simple way to augment imitation learning policies with a critic learned via expectile regression to effectively make use of non-expert data for recovery and broad generalization, even in the low data-coverage regime.

V. Experimental Setup

Evaluation Tasks: We investigate the RISE approach on manipulation tasks in simulation from the Robomimic benchmark [44], [45], and real-world robot tasks from the Furniture Bench [46] benchmark (as shown in Fig 4). We choose tasks that cover a range of characteristics including $\mathbb{SE}(2)$ object rearrangement (lampshade), $\mathbb{SE}(3)$ object manipulation (square-peg, piece-assembly), fine-precision (threading, cloth folding), and long-horizon behavior (mug-cleanup, one-leg). This work is evaluated on the Franka Panda robot both in simulation and the real world. In all evaluations, the policies and value functions receive camera images (from both wrist and third person cameras), as well as proprioceptive joint state from the robot. We refer the reader to the Appendix for task/implementation details.

Evaluation Settings: We consider three evaluation "settings" - (1) learning to recover from unstructured play data, (2) leveraging suboptimal failure data to improve success rates, and (3) iteratively improving a policy by finetuning on its own evaluation rollouts. For each task, we collect a set of expert demonstrations completing the task from a range of initial object configurations, and a set of non-expert demonstrations, which have different qualities for each setting, as we outline below. See Fig 9 for a visualization of the training data.

(1) *Learning to recover from unstructured play data:* This involves scenarios where a set of expert data that completes the task is augmented with a larger set of human collected, unstructured, undirected "play" data. This data demonstrates how to move the object around in the environment, thereby enabling recovery from unfamiliar starting conditions. For instance, in Fig 2, while the expert (shown in green) can only succeed from a narrow region, the undirected data (shown in yellow) can enable recovery back to this region to solve the task reliably across the state space. This type of data is typically very cheap to collect, as it does not require the precision to complete a task.

(2) *Leveraging suboptimal failure data:* This involves scenarios where a set of expert data that completes the task is augmented with a larger set of human collected suboptimal or failed demonstrations, which often occurs naturally during data collection. While the failed demonstrations are not suitable for direct imitation, they can still demonstrate useful subcomponents of the task. When these are stitched together with expert behavior, this leads to robust, higher-coverage policies, without wasting the entirety of the suboptimal data.

(3) *Iterative Policy Improvement:* We also demonstrate that useful non-expert data can be collected from policy rollouts, not just human demonstrations. Like in setting (1), given an initial expert dataset $\mathcal{D}_E$ and non-expert dataset $\mathcal{D}_{NE}$, we train a policy π^* as given in Equation 3. We then evaluate the policy π^*, and add successful rollouts to $\mathcal{D}_E$ and failed rollouts to $\mathcal{D}_{NE}$, then re-train.

We consider several imitation and offline RL baselines - (1) *Behavior cloning:* This is the standard imitation learning paradigm, with a diffusion policy [1] trained on only the expert dataset $\mathcal{D}_{\text{E}}$, (2) *Behavior cloning unified:* This is similar to behavior cloning, but on the union of expert and non-expert data $\mathcal{D}_{\text{E}} \cup \mathcal{D}_{\text{NE}}$, (3) *ILID:* This is an implementation of the data filtering algorithm in [36], where a classifier is used to classify expert vs non-expert states and subtrajectories that have overlap with expert data are selectively added to the training dataset for imitation learning, (4) *SQIL*: an online RL method that originally proposed 0/1 rewards, implemented as offline SAC [33], (5) *CQL*: a common offline RL method that enforces conservatism on the Q function [8], and (6) *IDQL*: the method RISE builds off of, without any data augmentation or Lipschitz penalty [10]. We modify the original IDQL implementation to use the $0/1$ rewards used in RISE.

VI. Experimental Results

a) RISE solves tasks from a broad range of initial configurations using high-coverage play data:

With the addition of low collection cost play data (as shown in Fig 2) to just expert data, our results indicate that RISE is able to achieve strong performance on a much wider distribution of initial configurations than an expert policy naively trained with imitation learning (Figure 5a). This can be seen from the improvement of RISE over BC in both simulation and real (See Coverage section in Table I). Crucially, we do not have to demonstrate expert behavior from this wider distribution, but simply collect enough coverage

Data Type	Sim Task Variant	BC	BCU	ILID	SQIL	CQL	IDQL	RISE
Coverage	square-peg	18.7 ± 2.4	0.0 ± 0.0	35.3 ± 3.5	0.0 ± 0.0	12.4 ± 3.2	19.6 ± 4.3	$\mathbf{50.7 \pm 5.8}$
	square-hook	18.0 ± 3.5	0.0 ± 0.0	34.6 ± 2.4	0.0 ± 0.0	10.5 ± 3.9	17.8 ± 5.8	$\mathbf{47.9 \pm 1.2}$
	piece-assembly	14.7 ± 2.9	2.0 ± 1.2	43.3 ± 2.4	3.3 ± 2.4	8.2 ± 1.5	16.3 ± 2.0	$\mathbf{70.7 \pm 8.8}$
	piece-assembly (tip)	0.0 ± 0.0	0.0 ± 0.0	9.3 ± 1.3	0.0 ± 0.0	0.0 ± 0.0	8.0 ± 2.3	$\mathbf{51.3 \pm 9.3}$
	threading	17.3 ± 2.6	0.0 ± 0.0	20.3 ± 1.9	0.0 ± 0.0	0.0 ± 0.0	9.8 ± 3.9	$\mathbf{22.7 \pm 1.4}$
Suboptimal	mug-cleanup	31.3 ± 3.5	32.7 ± 1.8	24.7 ± 4.1	6.0 ± 1.3	22.3 ± 2.0	36.7 ± 3.2	$\mathbf{40.7 \pm 5.3}$
	piece-assembly (tip)	20.0 ± 3.1	23.3 ± 5.7	22.7 ± 2.4	0.0 ± 0.0	16.7 ± 2.1	$\mathbf{35.7 \pm 4.5}$	$\mathbf{36.0 \pm 6.1}$
	square-peg	8.0 ± 2.3	34.0 ± 2.0	32.0 ± 2.3	8.3 ± 1.7	25.3 ± 2.2	41.3 ± 8.2	$\mathbf{56 \pm 2.3}$
Data Type	**Real Task Variant**	**BC**	**BCU**	**ILID**	**SQIL**	**CQL**	**IDQL**	**RISE**
Coverage	lampshade	17.5	45.0	57.5	0.0	0.0	10.0	**82.5**
	cloth folding	0.0	8.0	12.0	0.0	0.0	16.0	**24.0**
Suboptimal	one-leg	25.0	0.0	30.0	0.0	0.0	0.0	**50.0**

TABLE I: **Sim & real tasks across benchmarks:** Success percentage for an array of tasks with different types of human collected non-expert data. square-peg and square-hook share the same non-expert data.. Coverage refers to experiments utilizing high coverage play data (setting (1)), while suboptimal refers to experiments utilizing suboptimal failure data (setting (2)).

data which can be *stitched* with the expert data to enable recovery to the expert manifold. The BCU results suggest that simply imitating the high-coverage play data is insufficient, and this needs to be used in a targeted way. While ILID [36], along with the other offline RL baselines (SQIL, CQL, IDQL), can utilize suboptimal data to some extent, they are poor at stitching trajectories together, making them far less effective than RISE across tasks. This gap is particularly pronounced for the piece-assembly with tipping task, which requires combining multiple behaviors together (first rotating the object, then recovering). Notably, these results hold across both simulation and real world tasks. Moreover, since the non-expert data is simply used to recover back to the expert manifold, the same non-expert data can be useful across multiple downstream tasks. In Table I, RISE achieves good perfomance on the square-hook and square-peg tasks, which share the same non-expert data. This shows that the same data can be stitched to two different experts, offering a scalable way of improving policy robustness.

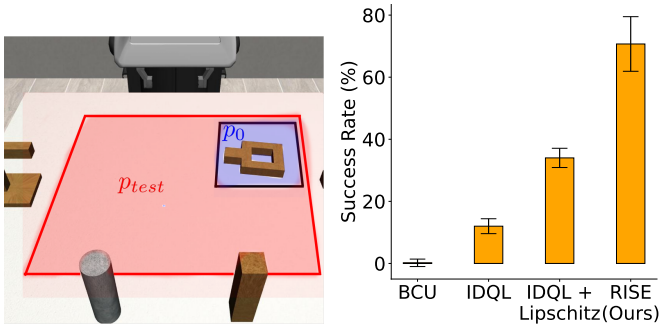


Fig. 5: **Generalization and Ablation** (a) All experts are demonstrated from a narrow initial distribution p_0. We test in a larger region p_{test}. Our method is able to generalize to p_{test} only using cheap play data. (b) Ablation of applying spectral norm regularization and data augmentation to standard IDQL on piece-assembly with high coverage. Standard offline RL (IDQL) does not stitch well, but adding our lipschitz constraint and data augmentation greatly improves performance.

b) *RISE is able to use suboptimal or partial data to improve policy performance:*

Our results show that RISE is able to utilize suboptimal or partial trajectory data to improve evaluation performance of the resulting policy (Table I under the Suboptimal data type section). While simply imitating a mixture of suboptimal data and optimal data leads to a considerable drop off in imitation learning methods (BCU), RISE is able to filter out the suboptimal data and do significantly better. Moreover, we see that RISE is actually able to outperform the standard BC baseline, which is simply imitating the expert data (while discarding suboptimal data). This suggests that RISE is not only filtering the data to ignore poor demonstrations, but also stitching suboptimal with optimal data to see additional benefit. As before, ILID [36] can show some benefit, but generally does not make maximal use of the suboptimal data because of its inability to stitch together data.

c) *RISE is able to leverage data collected from policy evaluations:*

Policy evaluations are run frequently in the real world, and provide a rich source of additional data. While not typically used in the learning pipeline, we show that iteratively re-integrating this evaluation data into policy learning can help. Table II shows that RISE is able to leverage data collected from policy evaluation to improve policy performance without any additional human demonstrations. Given an initial policy that performs relatively poorly at the task, we are able to use the data from the rollouts from that very same policy to improve the policy by categorizing them as either successful trajectories or failures. With each subsequent round of data collection and re-training, we see that the policy performance increases. This demonstrates the versatility of RISE in utilizing all forms of non-expert data.

Task	Initial Performance	Iteration 1	Iteration 2
piece-assembly	26.3	42.7	49.0
lampshade	20.0	55.0	60.0

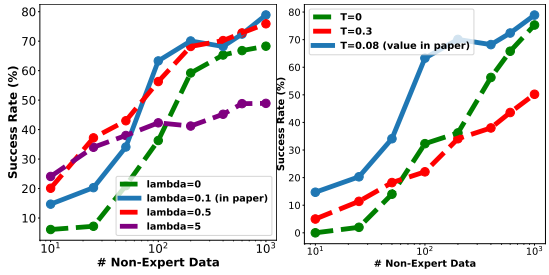
TABLE II: Results for iterative policy improvement using data collected autonomously from policy rollouts. Given a poor initial policy, additional data is collected from its rollouts to finetune the policy. This process is repeated over multiple iterations.

d) *Ablations and Analysis*

Impact of Lipschitz continuity and Data Augmentation: To understand the impact of imposing Lipschitz continuity on RISE and data augmentation, we also perform a targeted ablation on the piece-assembly task in simulation. From Fig 5b, we can see that offline RL for recovery (without any smoothness additions) performs better than naively doing BC, but can be improved by imposing of Lipschitz continuity through the spectral norm. Fig. 5b further shows performance gains by adding the distance-based data augmentation.

Impact of smoothing hyperparameters: We examine the ef-

fect of various parameters of λ, the strength of the spectral norm regularization, T, the distance threshold governing the degree of data augmentation, and $|\mathcal{D}_{NE}|$, the amount of non-expert data. In, Figure 6 (a) and (b), we see the sensitivity of RISE with respect to λ and T, respectively on the `piece-assembly` task, and its relation to the amount of data. We see that a moderate amount of spectral normalization and data augmentation greatly increases policy success, and as expected, this improvement is greatest when data is limited. The performance is somewhat sensitive to hyperparameter values, but a large range of values is beneficial.



(a) (b)

Fig. 6: Ablations on relation between data quantity and (a) λ and (b) T hyperparameters for the `piece-assembly` task. Moderate spectral normalization penalty and data augmentation, which expands the policy's action distribution, is critical, particularly when data is scarce.

VII. Conclusion and Limitations

`RISE` provides a new way to use ideas from offline RL to improve the robustness of imitation learning, but without requiring the challenging reward labeling procedure involved in most offline RL methods. Informally, key insight here is to "fuzz" the dataset in places where precision is not required using a notion of Lipschitz continuity. With this, however, comes a caveat: You must know which parts of the dataset needs to be precise and which parts can sacrifice precision for stitch-ability. In some settings, it is clear, while in others this may require more careful tuning. We also find that there are scenarios where the suboptimal and optimal data do not overlap, despite the smoothing offered by Lipschitz continuity and data augmentation. A clear understanding of what data sources will yield benefits would be valuable in future studies.

VIII. Acknowledgements

The authors would like to acknowledge members of the Robot Learning Lab and the Washington Embodied Intelligence and Robotics Development Lab for helpful and informative discussions throughout the process of this research. The authors would also like to thank Emma Romig at the University of Washington for their help in setting up the robotic hardware and teleoperation interfaces for this project. Toyota Research Institute provided funds to support this work.

References

[1] C. Chi, S. Feng, Y. Du, Z. Xu, E. Cousineau, B. Burchfiel, and S. Song, "Diffusion policy: Visuomotor policy learning via action diffusion," in *Robotics: Science and Systems*, 2023.

[2] T. Z. Zhao, V. Kumar, S. Levine, and C. Finn, "Learning fine-grained bimanual manipulation with low-cost hardware," in *Robotics: Science and Systems*, 2023.

[3] M. Zare, P. M. Kebria, A. Khosravi, and S. Nahavandi, "A survey of imitation learning: Algorithms, recent developments, and challenges," 2023.

[4] A. Majumdar, M. Sharma, D. Kalashnikov, S. Singh, P. Sermanet, and V. Sindhwani, "Predictive red teaming: Breaking policies without breaking robots," *CoRR*, vol. abs/2502.06575, 2025.

[5] S. Belkhale, Y. Cui, and D. Sadigh, "Data quality in imitation learning," in *NeurIPS*, 2023.

[6] J. Hejna, S. Mirchandani, A. Balakrishna, A. Xie, A. Wahid, J. Tompson, P. Sanketi, D. Shah, C. Devin, and D. Sadigh, "Robot data curation with mutual information estimators," *CoRR*, vol. abs/2502.08623, 2025.

[7] C. Agia, R. Sinha, J. Yang, R. Antonova, M. Pavone, H. Nishimura, M. Itkina, and J. Bohg, "CUPID: curating data your robot loves with influence functions," *CoRR*, vol. abs/2506.19121, 2025.

[8] A. Kumar, A. Zhou, G. Tucker, and S. Levine, "Conservative q-learning for offline reinforcement learning," in *Advances in Neural Information Processing Systems 33* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), 2020.

[9] Y. Wu, G. Tucker, and O. Nachum, "Behavior regularized offline reinforcement learning," *CoRR*, vol. abs/1911.11361, 2019.

[10] P. Hansen-Estruch, I. Kostrikov, M. Janner, J. G. Kuba, and S. Levine, "IDQL: implicit q-learning as an actor-critic method with diffusion policies," *CoRR*, vol. abs/2304.10573, 2023.

[11] R. S. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. MIT Press, 2 ed., 2018.

[12] A. Hussein, M. M. Gaber, E. Elyan, and C. Jayne, "Imitation learning: A survey of learning methods," *ACM Computing Surveys (CSUR)*, 2017.

[13] T. Osa, J. Pajarinen, G. Neumann, J. A. Bagnell, P. Abbeel, and J. Peters, "An algorithmic perspective on imitation learning," *Foundations and Trends in Robotics*, 2018.

[14] B. D. Argall, S. Chernova, M. Veloso, and B. Browning, "A survey of robot learning from demonstration," in *Robotics and autonomous systems*, vol. 57, pp. 469–483, Elsevier, 2009.

[15] H. Ravichandar, A. S. Polydoros, S. Chernova, and A. Billard, "Recent advances in robot learning from demonstration," *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 3, pp. 297–330, 2020.

[16] S. Ross, G. J. Gordon, and J. A. Bagnell, "A reduction of imitation learning and structured prediction to no-regret online learning," 2011.

[17] L. Ke, Y. Zhang, A. Deshpande, S. Srinivasa, and A. Gupta, "Ccil: Continuity-based data augmentation for corrective imitation learning," 2024.

[18] L. Ke, S. Choudhury, M. Barnes, W. Sun, G. Lee, and S. Srinivasa, "Imitation learning as f-divergence minimization," 2020.

[19] X. Zhang, M. Chang, P. Kumar, and S. Gupta, "Diffusion meets dagger: Supercharging eye-in-hand imitation learning," 2024.

[20] M. Laskey, J. Lee, R. Fox, A. Dragan, and K. Goldberg, "Dart: Noise injection for robust imitation learning," 2017.

[21] N. M. Shafiullah, Z. J. Cui, A. Altanzaya, and L. Pinto, "Behavior transformers: Cloning k modes with one stone," in *Advances in Neural Information Processing Systems 35* (S. Koyejo, S. Mohamed, A. Agarwal, D. Belgrave, K. Cho, and A. Oh, eds.), 2022.

[22] T. Z. Zhao, J. Tompson, D. Driess, P. Florence, S. K. S. Ghasemipour, C. Finn, and A. Wahid, "Aloha unleashed: A simple recipe for robot dexterity," in *Proceedings of The 8th Conference on Robot Learning*, vol. 270 of *Proceedings of Machine Learning Research*, pp. 1910–1924, PMLR, 06–09 Nov 2025.

[23] K. Black, N. Brown, D. Driess, A. Esmail, M. Equi, C. Finn, N. Fusai, L. Groom, K. Hausman, B. Ichter, S. Jakubczak, T. Jones, L. Ke, S. Levine, A. Li-Bell, M. Mothukuri, S. Nair, K. Pertsch, L. X. Shi, J. Tanner, Q. Vuong, A. Walling, H. Wang, and U. Zhilinsky, "π_0: A vision-language-action flow model for general robot control," 2024.

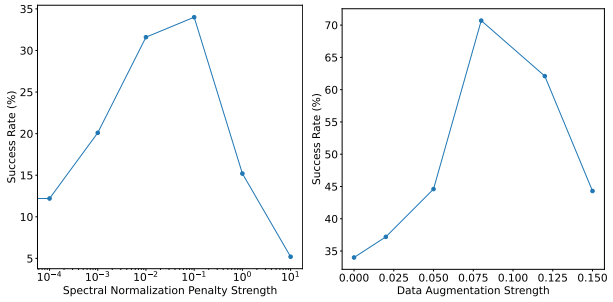
[24] A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, and B. Zitkovich, "Rt-2: Vision-language-action models transfer web knowledge to robotic control," 2023.

[25] M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar,

B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, and C. Finn, "Openvla: An open-source vision-language-action model," 2024.
[26] S. Karamcheti, S. Nair, A. S. Chen, T. Kollar, C. Finn, D. Sadigh, and P. Liang, "Language-driven representation learning for robotics," in *Robotics: Science and Systems (RSS)*, 2023.
[27] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," 2020.
[28] N. Jiang and L. Li, "Doubly robust off-policy value evaluation for reinforcement learning," in *Proceedings of the 33nd International Conference on Machine Learning*.
[29] Z. Fang and T. Lan, "Learning from random demonstrations: Offline reinforcement learning with importance-sampled diffusion models," *CoRR*, vol. abs/2405.19878, 2024.
[30] T. Yu, G. Thomas, L. Yu, S. Ermon, J. Y. Zou, S. Levine, C. Finn, and T. Ma, "MOPO: model-based offline policy optimization," in *Advances in Neural Information Processing Systems 33* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), 2020.
[31] R. Kidambi, A. Rajeswaran, P. Netrapalli, and T. Joachims, "Morel: Model-based offline reinforcement learning," in *Advances in Neural Information Processing Systems 33* (H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, eds.), 2020.
[32] I. Kostrikov, A. Nair, and S. Levine, "Offline reinforcement learning with implicit q-learning," 2021.
[33] S. Reddy, A. D. Dragan, and S. Levine, "Sqil: Imitation learning via reinforcement learning with sparse rewards," 2019.
[34] G. J. Gao, T. Li, and N. Figueroa, "Out-of-distribution recovery with object-centric keypoint inverse policy for visuomotor imitation learning," 2025.
[35] A. Reichlin, G. L. Marchetti, H. Yin, A. Ghadirzadeh, and D. Kragic, "Back to the manifold: Recovering from out-of-distribution states," 2022.
[36] S. Yue, J. Liu, X. Hua, J. Ren, S. Lin, J. Zhang, and Y. Zhang, "How to leverage diverse demonstrations in offline imitation learning," 2024.
[37] A. Singh, H. Liu, G. Zhou, A. Yu, N. Rhinehart, and S. Levine, "Parrot: Data-driven behavioral priors for reinforcement learning," in *9th International Conference on Learning Representations, ICLR*, 2021.
[38] S. Levine, A. Kumar, G. Tucker, and J. Fu, "Offline reinforcement learning: Tutorial, review, and perspectives on open problems," *CoRR*, vol. abs/2005.01643, 2020.
[39] J. Ho, A. Jain, and P. Abbeel, "Denoising diffusion probabilistic models," in *Advances in Neural Information Processing Systems 33*.
[40] S. Park, K. Frans, S. Levine, and A. Kumar, "Is value learning really the main bottleneck in offline rl?," 2024.
[41] Y. Yoshida and T. Miyato, "Spectral norm regularization for improving the generalizability of deep learning," *CoRR*, vol. abs/1705.10941, 2017.
[42] M. O'Connell, G. Shi, X. Shi, K. Azizzadenesheli, A. Anandkumar, Y. Yue, and S.-J. Chung, "Neural-fly enables rapid learning for agile flight in strong winds," *Science Robotics*, vol. 7, May 2022.
[43] M. Oquab, T. Darcet, T. Moutakanni, H. Vo, M. Szafraniec, V. Khalidov, P. Fernandez, D. Haziza, F. Massa, A. El-Nouby, M. Assran, N. Ballas, W. Galuba, R. Howes, P.-Y. Huang, S.-W. Li, I. Misra, M. Rabbat, V. Sharma, G. Synnaeve, H. Xu, H. Jegou, J. Mairal, P. Labatut, A. Joulin, and P. Bojanowski, "Dinov2: Learning robust visual features without supervision," 2024.
[44] A. Mandlekar, D. Xu, J. Wong, S. Nasiriany, C. Wang, R. Kulkarni, L. Fei-Fei, S. Savarese, Y. Zhu, and R. Martín-Martín, "What matters in learning from offline human demonstrations for robot manipulation," 2021.
[45] A. Mandlekar, S. Nasiriany, B. Wen, I. Akinola, Y. Narang, L. Fan, Y. Zhu, and D. Fox, "Mimicgen: A data generation system for scalable robot learning using human demonstrations," 2023.
[46] M. Heo, Y. Lee, D. Lee, and J. J. Lim, "Furniturebench: Reproducible real-world benchmark for long-horizon complex manipulation," in *Robotics: Science and Systems*, 2023.
[47] E. Todorov, T. Erez, and Y. Tassa, "Mujoco: A physics engine for model-based control," in *2012 IEEE/RSJ international conference on intelligent robots and systems*, pp. 5026–5033, IEEE, 2012.
[48] P. Wu, Y. Shentu, Z. Yi, X. Lin, and P. Abbeel, "Gello: A general, low-cost, and intuitive teleoperation framework for robot manipulators," 2023.
[49] Franka Emika, "libfranka."
[50] Tim Schneider, "franky: High-level motion library for the franka emika robot."
[51] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

APPENDIX

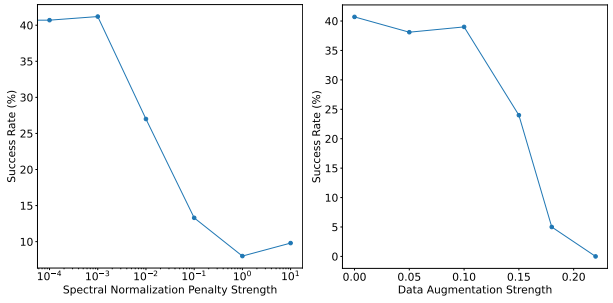
Additional Ablations We provide a few more examples of ablations into the hyperparameters λ and T. First, looking at the `piece-assembly` task, we see that a moderate amount of spectral normalization and data augmentation greatly increases policy success, as seen in Figure 7. From this ablation, we use the best value of λ ($\lambda = 0.1$) for all experiments utilizing high coverage non-expert datasets in simulation. Data augmentation strength, however, has to be tuned per-task.



(a) (b)

Fig. 7: **Ablation on piece assembly task** Effect of various (a) spectral normalization penalty strength parameters λ and (b) data augmentation threshold parameters T on the piece assembly task. Spectral normalization is applied assuming no data augmentation, while data augmentation ablations are done using the optimal level of spectral normalization $\lambda = 0.1$

We use the `mug-cleanup` task as representative of experiments where we use suboptimal and partial demonstrations. For these datasets, we use the best value of $\lambda = 0.001$ demonstrated in Figure 8.



(a) (b)

Fig. 8: **Ablation on mug cleanup task** Effect of various (a) spectral normalization penalty strength parameters λ and (b) data augmentation threshold parameters T on the mug cleanup task. Spectral normalization is applied assuming no data augmentation, while data augmentation ablations are done using the optimal level of spectral normalization $\lambda = 0.001$

Experimental Setup All simulation task environments are modified from versions implemented in Robomimic or Mimicgen [44], [45], which are built on the Mujoco simulator [47]. Data is collected with a SpaceMouse device. Real experiments are conducted on a Franka Panda robot. We collect robot trajectory demonstrations using the Gello leader arm [48]. During collection, the Panda follows them using the waypoint-tracking impedence controller provided by `libfranka` [49] and the `franky` wrapper [50]. In both simulation and real, the action space consists of delta cartesian end-effector commands. The observation consists of proprioception from the

robot, in the form of the cartesian end effector position, the orientation of the end effector as a quaternion, and the gripper state. The observation also contains two RGB images, one from a fixed exocentric camera, and one wrist mounted camera. In real, the camera images are captured from two Intel Realsense D435 cameras.

Architecture and Training The behavior policy is parametrized as a diffusion policy with a U-Net denoiser, with the same implementation and parameters as [1]. The value network and Q network are each a 3-layer MLP. We use the ResNet18 architecture [51] for the image encoder, which is trained jointly with the behavior policy and Q function. Like in [1], we use *action chunking*, where the policy predicts a sequence of actions [2]. The behavior policy also takes as input an observation history, where the observations from the last two timesteps are stacked together. For all tasks, each network is trained for 5000 epochs.

Data Augmentation For computational efficiency, we approximate the data augmentation procedure described in Section IV-C using a k-nearest neighbors algorithm. For each state, we compute the $k = 15$ nearest neighbors under the distance metric d. If the distance is less than the threshold T, and the two states belong to different trajectories, we accept the the states, swapping their actions. To compute the distance metric, we use the ViT-S model from DinoV2 [43], extract the patch tokens, normalize, and then compute euclidean distance.

Hyperparameters Hyperparameters used are provided in Table III. Data augmentation strength T is tuned per-task. Data quantities for each task is found in Table IV.

Parameter	Value
Learning Rate (all)	1e-4
Batch Size	64
Diffusion Timesteps	100
Beta Schedule	cosine
Discount Factor γ	0.99
IQL Expectile τ	0.9
# Samples from Behavior Policy	64
Spectral norm regularization strength λ	0.1 (coverage experiments) 0.001 (suboptimal experiments) 0.01 (real)
Action chunking horizon	16
Observation history horizon	2

TABLE III: Table of hyperparameters used.

We provide a description of each task and associated datasets. A summary of the data used for each task can be found in Table IV.

`square-peg` The task involves picking up a square nut and placing it over a peg. In the expert data, the square nut is initialized in a small region p_0 on the upper right of the table. For the "high-coverage play" data, the nut is randomized across the entire table p_{test}, and randomly picked up and moved around in 3D space. The "suboptimal" data is a subset of Robomimic's "Square-Worse" dataset for the square task [44], where an inexperienced human demonstrator attempts to complete the task. For the coverage experiment, we evaluate with the nut initialized to p_{test}. For the suboptimal experiment, we evaluate with the nut initialized from p_0.

`square-hook` The task exists in the same environment as `square-peg`, except the goal is to hang the square nut onto a hook. The initial distribution of the nut in the expert data is also identical to `square-peg`. We re-use the same high coverage data as the `square-peg` task to demonstrate the scalability of this type of data to augment multiple downstream tasks.

`piece-assembly` This task involves picking up a T-shaped block and placing it into another square-shaped block. In the expert data, the block is initialized in a narrow region p_0 in the top right. In the "high-coverage play" data, the block is initialized across the entire table p_{test}, and randomly picked up and moved around. In the "suboptimal" demonstrations, the block is also initialized in p_0, expert behavior is attempted, but is either executed poorly or unsuccessful (see Figure 9). In the tasks that include tipping, the block is initialized on its side, and must be re-oriented to be upright before being picked up. The "partial tipping" non-expert data, the block is reoriented using a variety of tipping strategies. For the coverage experiment, we evaluate with the block initialized to p_{test}. For the suboptimal experiment, we evaluate with the block initialized from p_0.

`threading` This task involves threading a needle-like object into a small hole, requiring precision. The expert region p_0 is a narrow region on the right of the table. In the "high-coverage play" data, the needle is initialized across the entire table p_{test}, and randomly picked up and moved around.

`mug-cleanup` This task involves opening a drawer, picking up a mug, and placing it into the drawer. Suboptimal data includes demonstrations where the demonstrator only opens the drawer, but then fails at grabbing the mug, and demonstrations where the drawer starts open, and the demonstrator places the mug in the drawer without demonstrating opening the drawer (see Figure 10). In all cases, the mug is initialized from a small region p_0, which also serves as the evaluation region.

`lampshade` This task involves picking up a lampshade and placing it on a partially assembled stand. The expert is collected from a narrow region p_0, while the "high-coverage data" is collected from a wider initialization region p_{test}, where the lampshade is randomly pushed around the table.

`one-leg` This task involves picking up a table leg and inserting it into a hole of a table. The expert is collected from a narrow region p_0. The suboptimal data consists of picking the leg from a wide range p_{init}, but failing to insert the leg into the hole successfully.

`cloth folding` This task involves folding a piece of cloth and then stacking it. The high coverage play involves randomly re-arranging and folding the cloth from various starting configurations. The expert data consists of picking up the cloth and stacking it from a narrow initial distribution where the cloth is already neatly placed.

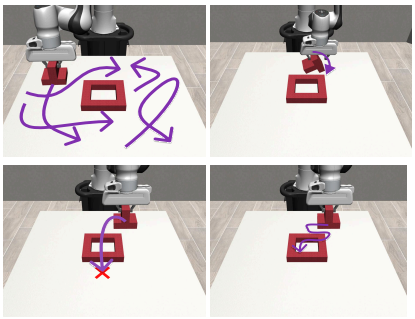
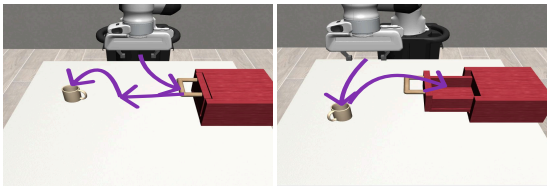


Fig. 9: Visualizations of types of play data for the `piece-assembly` task. (top left) high coverage play data — the block is randomly picked up and moved around the table (top right) partial tipping — the block is tipped over to its upright position (bottom left) failure — the block is attempted to be inserted, but misses (bottom right) suboptimal — the block is inserted into the hole, but in an inefficient manner



(a) (b)

Fig. 10: Visualization of non-expert data for the `mug-cleanup` task (a) drawer open only — the demonstration only consists of opening the drawer and attempting to grasp the mug (b) place in drawer only — the demonstration only consists of picking the mug up and placing it in the drawer

Task	Dataset	Data Augmentation Threshold (T)
`square-peg` (Coverage)	Expert (200) High Coverage Play (224)	0.15
`square-hook` (Coverage)	Expert (175) High Coverage Play (224)	0.15
`piece-assembly` (Coverage)	Expert (200) High Coverage Play (199)	0.08
`piece-assembly` (tipping) (Coverage)	Expert (200) High Coverage Play (199) Partial Tipping (104)	0.08
`threading` (Coverage)	Expert (200) High Coverage Play (200)	0.08
`mug-cleanup` (Suboptimal)	Expert (20) Suboptimal (85)	0.05
`square-peg` (Suboptimal)	Expert (20) Suboptimal (80)	0.0
`piece-assembly` (tipping) (Suboptimal)	Expert (10) Suboptimal (100)	0.0
`lampshade`	Expert (225) High Coverage Play (276)	0.22
`one-leg`	Expert (50) Suboptimal (150)	0.1
`cloth folding`	Expert (50) High Coverage Play (50)	0.05

TABLE IV: Composition of datasets used for each task, along with the amount of data augmentation used. The number in parenthesis indicates the number of trajectories in that dataset.