Beyond the Teacher: Leveraging Mixed-Skill Demonstrations for Robust Imitation Learning

Abstract: Achieving expert-like robotic task execution in dynamic environments typically requires extensive, high-quality expert demonstrations, a significant bottleneck for real-world deployment. We present a novel learning framework that overcomes this data dependency, enabling robots to perform complex periodic tasks with expert-like proficiency, even when learning from naive demonstrations. Our two-stage pipeline first selects a representative demonstration based on user-defined information-aware task intention scores. This single best demo is then used to extract a canonical motion shape via Periodic Dynamic Movement Primitives (DMPs). Finally, a Long Short-Term Memory (LSTM) network refines the entire set of demonstrations, leveraging a multi-objective score that combines the canonical shape with mutual information and other task quality metrics. The proposed approach is demonstrated on a Franka Research 3 robot performing phasic tasks across three contrasting domains: wiping and stirring in human assistive services, weaving in the textile industry, and pick-and-place operations for warehouse automation. Code available at: https://github.com/codesudopaper/MSDR2025.

1 Introduction

Imitation Learning (IL) enables robots to acquire complex behaviors by learning state-action mappings from expert demonstrations, bypassing the need for hand-crafted reward functions or extensive environment exploration [1]. However, a major limitation of conventional IL methods is their dependence on large quantities of high-quality expert demonstrations. When demonstrations are suboptimal, inconsistent or scarce, as is often the case in real-world robotic settings, learned policies suffer from compounding errors due to covariate shift, and scalability becomes a bottleneck due to high data acquisition costs [2].

To address this, we propose a novel IL pipeline that bypasses the reliance on abundant expert data. Instead, our method recovers a high-quality, expert-like dataset from a single clean demonstration using a combination of periodic Dynamic Movement Primitives (DMPs) and LSTM-based trajectory sequence correction. This recovered dataset enables robust policy learning even when the remaining data is noisy, suboptimal, or corrupted.

Recent works such as Making Imitation Learning Easy with Self-Supervision (MILES) [3], Co-Imitation Learning (CoIL) [4], and Task-Oriented Self-Imitation Learning (TOSIL) [5] mitigate the disadvantage of poor demonstrations by using self-generated data through good or bad experiences or human interventions. Others, like ILEED [6] and [7], estimate the ability of the demonstrator or trajectory feasibility to filter or weight demonstrations for behavior cloning. However, these methods typically require access to large demonstration datasets and sufficient exploration of the state space to effectively learn latent representations and struggle with complex, highly structured behaviors, such as periodic motions with high curvature, as seen in handwriting or stirring tasks.

Data curation strategies for imitation learning under imperfect demonstrations, such as Zhang et al. [8] present Self-Supervised Approach to Data Curation for Large-Scale Imitation Learning (SCIZOR), a framework for large-scale curation that removes redundant or suboptimal samples, though it does not explicitly measure the information content of state–action pairs, risking the loss of task-

9th Conference on Robot Learning (CoRL 2025), Seoul, Korea..

relevant features. Belkhale et al. [9] analyze data quality by characterizing diversity and action consistency as key factors for generalization, yet lack an online mechanism to score samples. Du et al. [10] address data scarcity by proposing behavior retrieval from unlabeled datasets, which relies on the availability of diverse priors and suitable retrieval metrics. Nguyen et al. [11] show that leveraging fully observable policies can improve learning under partial observability, emphasizing the importance of preserving predictive information. Hejna et al. [12] take a more direct approach by employing mutual information estimators to quantify the dependence between states and actions, demonstrating that higher mutual information correlates with better imitation performance.

2 Preliminaries and Problem Formulation

We aim to learn high-curvature periodic motion tasks, such as wiping, stirring, weaving, and pick-and-place tasks on an n-DoF robotic manipulator (in this study, we use the 7-DoF Franka Research 3 Panda robot [13]), where the end effector (EE) exhibits structured rhythmic behaviors in Cartesian space X. Mathematically, the input to our pipeline is a collection of M demonstration trajectories:

$$E = \left\{ \{x_i(t_k)\}_{k=1}^{N} \right\}_{i=1}^{M}, \quad x_i(t_k) \in \mathbb{R}^d,$$

where $x_i(t_k)$ denotes the Cartesian position of the end effector at time t_k in the i-th demonstration, and each trajectory contains N samples. The goal is to learn the continuous-time motion dynamics:

$$\dot{x} = f(x), \quad x \in X \subset \mathbb{R}^d, \tag{1}$$

where d can be 2 or 3 depending on 2D or 3D cartesian space for EE, $f : \mathbb{R}^d \to \mathbb{R}^d$ defines a nominal vector field that governs the end effector dynamics. This vector field corresponds to the Cartesian velocity of the EE, which is obtained via forward kinematics from the robot's joint velocities, tracked using a low-level impedance-based controller. In practice, however, the demonstration set E can

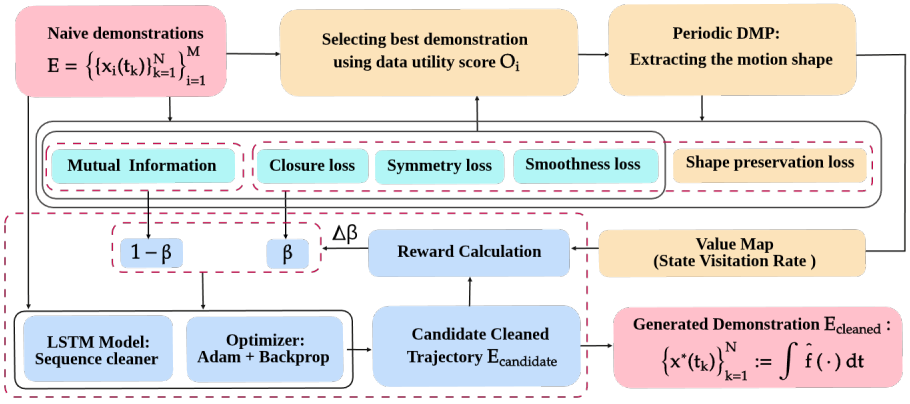


Figure 1: Control flow diagram of the proposed approach. Pink blocks represent the set of demonstrations, yellow blocks represent the first step in the strategy to extract the required task motion pattern and value map on task space, blue blocks represent the second step in the strategy to correct the naive demonstrations with adaptive β, and cyan blocks represent the mutual information and multi-objective loss functions from task intention.

include noisy, suboptimal, or inconsistent trajectories that degrade the learned policy $\hat{f}_E$ obtained through standard Imitation Learning (IL) methods [14], [15], [16]. Our goal is to synthesize a corrected dataset E_{cleaned} such that

$$E_{\text{cleaned}} = \left\{ \{\hat{x}_i(t_k)\}_{k=1}^{N} \right\}_{i=1}^{M},$$

and the resulting learned policy $\hat{f}_{\text{cleaned}}$, generates trajectories through roll out that maximize the mutual information between random variables for the state sequence $S(x_i) = \{x_i(t_k)\}_{k=1}^{N}$ and action sequence $A(x_i) = \{a_i(t_k)\}_{k=1}^{N}, a_i(t_k) = x_i(t_{k+1}) - x_i(t_k), a_N(t_k) = 0$, denoted as $\mathcal{I}(S; A)$:

$$\mathcal{I}(S; A) = \mathcal{H}_S + \mathcal{H}_A - \mathcal{H}_{(S,A)}, \tag{2}$$

where entropy $\mathcal{H}_X$ is the measure of uncertainty for the random variable X. Furthermore, we need to minimize the user-defined task intention-based loss functions $L(x_i)$, which characterizes the required task features such as symmetry, task-specific shape, smoothness and closed-loop behavior more effectively than those generated by $\hat{f}_E$:

$$\mathbb{E}_{x\sim\text{Rollout}(\hat{f}_{\text{cleaned}})}[U(x)] > \mathbb{E}_{x\sim\text{Rollout}(\hat{f}_E)}[U(x)], \tag{3}$$

where $U(x) = (1-\beta)\mathcal{I}(S(x); A(x)) - \beta L(x)$ is the *demo utility score*, $\beta \in (0, 1)$ is the demonstrator ability's estimate (for expert $\beta = 1$), and $x \sim \text{Rollout}(\hat{f})$ denotes that the state trajectory $x(t)$, which is a solution of $\dot{x} = \hat{f}(x)$ starting from an initial condition. A higher demo utility score $U(x)$ improves behavior cloning by preserving informative state-action structure via mutual information (as shown by [12]) while enforcing task consistency through $L(x)$, thus reducing reliance on large datasets.

Unlike prior works, we do not assume access to a large set of high-quality expert demonstrations. Instead, we assume that the dataset contains a single expert-like trajectory $e^* \in E$. Using this, we construct a compact representation of the task via periodic Dynamic Movement Primitives (DMPs). A trajectory sequence-to-sequence correction model (e.g. LSTM) then maps noisy trajectories toward this expert-like reference, enabling policy learning from weak supervision with improved robustness and generalization. Moving forward, we will demonstrate the method for cleaning a dataset where EE is in a 2D Euclidean space, which can be easily generalized for a 3D Euclidean space for the end effector.

3 Proposed Approach

This section outlines a novel data-efficient hierarchical approach for robot motion planning (as shown in Fig. 1), integrating offline learning from suboptimal demonstrations.

3.1 Demo Utility Score: An Information-Aware Multi-objective Scoring Function

The raw set of demonstrations is often collected from teleoperation or crowd sourcing (regardless of expertise). These naive recordings are frequently of low and inconsistent quality, often presenting as noisy, inaccurate, or even trash data due to errors, incomplete tasks, or inefficient movements. Consequently, a correction strategy is designed to transform this raw input into a reliable input suitable for live robot operation using IL, using only a few demonstrations. Each demonstration trajectory for tasks is provided with general prior demonstrator preferences applied during the drawing of imperfect trajectories as the rotation and scaling parameter $\theta \in (-2\pi, 2\pi)$ and $\rho \in \mathbb{R}$, respectively. We employ a multi-objective scoring mechanism derived for task-intentional prompt, given by Do's and Don'ts of the task, easily accessible from the task manual. This quantitative evaluation allows us to prioritize the desired actions at a given state and correct the demonstrations of the entire raw dataset.

Demo utility score (U_i): $U(x_i)$ is calculated for each raw demonstration trajectory x_i in E. This score quantifies each demonstration's quality based on the mutual information between states and action as well as other important kinematic and geometric characteristics of the trajectory itself. Using the intrinsic characteristics of the sequence of pairs $(x_i(t_k), x_i(t_{k+1})), \forall k \geq 1$, we compute mutual information and various loss components; a higher mutual information and lower loss value denotes a more desirable characteristic. The demo utility score $U(x_i)$, is defined as a convex combination of mutual information $\mathcal{I}(S; A)$ and composite multi-objective loss function $-L(x_i)$ for N individual loss components, given the demonstrator i's ability estimate $\beta_i \in (0, 1)$:

$$\begin{aligned} U(x_i) &= (1-\beta_i)\mathcal{I}(S(x_i); A(x_i)) - \beta_i L(x_i), \\ L(x_i) &= \sum_j w_j \cdot L_j(x_i), \forall j \in \{1, 2, ..., N\}, \end{aligned} \tag{4}$$

where $L_j(x_i)$ is j-th loss component computed for the raw demonstration x_i, and $w_j \in [0, 1]$ is its corresponding weight.

Mutual information $\mathcal{I}(S;A)$ between state sequence $S(x_i)$ and action sequence $A(x_i)$ can be estimated using the Kraskov–Stögbauer–Grassberger (KSG) estimator [17] without explicit density modeling. The key steps are:

- For each sample, determine the distance to its k-nearest neighbor in the joint (S, A) space, which defines a local neighborhood radius.
- Count how many neighbors fall within this radius separately in the marginal S and A spaces, which act as proxies for marginal densities.
- Combine these counts using the digamma-based KSG formula, shown in Eq. (30) of [17], yielding an estimate of $\mathcal{I}(S;A) = \mathcal{H}_{S(x_i)} + \mathcal{H}_{A(x_i)} - \mathcal{H}_{(S(x_i),A(x_i))}$.

Intuitively, when the state sequence $S(x_i)$ and action sequence $A(x_i)$ are strongly dependent, the distance to the k-th nearest neighbor in the joint space (S, A) becomes smaller, indicating that state–action pairs cluster more tightly together. At the same time, the number of neighbors within the same joint radius is larger when viewed in the marginal spaces of S and A. In imitation learning, maximizing $\mathcal{I}(S;A)$ therefore favors demonstrations where states and actions are consistently related, leading to policies that are more robust under noisy or scarce data.

The specific formulations for these individual loss components are given as follows, assuming $x_i = \{p_{ik}, q_{ik}\}_{k=1}^{N}$ represents the trajectory point as a vector of 2D coordinates (e.g. $(p_{ik}, q_{ik}) \in \mathbb{R}^2$).

- *Symmetry Loss* (L_1): It quantifies the symmetry of a trajectory with respect to the y-axis. Let C_{ix} be the x-coordinate of the trajectory's centroid, $C_{ix} = \frac{1}{N}\sum_{k=1}^{N} p_{ik}$.

$$L_1(x_i) = \frac{1}{N}\sum_{k=1}^{N} \min_{j\in\{1,\dots,N\}} \lambda_i, \tag{5}$$
$$\lambda_i = \|(2C_{ix} - p_{ik}, q_{ik}) - (p_{ij}, q_{ij})\|_2,$$

  where $\|\cdot\|_2$ denotes Euclidean norm and it can be easily changed for symmetry across a given axis, by aligning the x-axis along the required axis.

- *Closure Loss* (L_2): It quantifies the distance between the start and end point of a trajectory, which creates a closed loop.

$$L_2(x_i) = \|(p_{i0}, q_{i0}) - (p_{iN}, q_{iN})\|_2, \tag{6}$$

  where $(p_{i0}, q_{i0}), (p_{iN}, q_{iN})$ is the start and end point of the given trajectory x_i.

- *Smoothness Loss* (L_3): The smoothness of a trajectory is evaluated using a weighted sum of the squared norms of its first, second, and third-order finite differences, which correspond to velocity, acceleration, and jerk, respectively. Let the trajectory be represented as a sequence of N points $\{x_i(t_k)\}_{k=1}^{N}$ in $\mathbb{R}^2$. Define D, D^2, and D^3 as first, second, and third-order difference matrices. The smoothness loss is computed as:

$$L_3(x_i) = \alpha_1\|D\Gamma_i\|_2^2 + \alpha_2\|D^2\Gamma_i\|_2^2 + \alpha_3\|D^3\Gamma_i\|_2^2, \tag{7}$$

  where $\Gamma_i \in \mathbb{R}^{N\times 2}$ is the trajectory, and α_1, α_2, and α_3 are weights (e.g. $\alpha_1 = 0.2$, $\alpha_2 = 0.3$, $\alpha_3 = 0.5$) that balance the contributions of velocity, acceleration, and jerk to the total loss.

Using min–max normalization, we rescale mutual information $\mathcal{I}(S;A)$ as $\tilde{\mathcal{I}}$ and each loss component $-L_j(x_i)$ as $(1 - \tilde{L}_j(x_i))$ to the [0,1] interval by subtracting the minimum value and dividing by the range, with small δ, for any metric I as

$$\tilde{I}_j = \begin{cases} \dfrac{I_j - \min_j I_j}{\max_j I_j - \min_j I_j}, & \text{if } \max_j I_j - \min_j I_j > \delta, \\ 0, & \text{otherwise,} \end{cases} \tag{8}$$

then the normalized demo utility score for the trajectory x_i, $\tilde{U}_i(x_i) \in [0,1]$ is computed as

$$\tilde{U}_i(x_i) = (1-\beta_i)\tilde{\mathcal{I}}(S(x_i); A(x_i)) + \beta_i \sum_j w_j (1 - \tilde{L}_j(x_i)). \tag{9}$$

For the initial demonstration selecearning Without Expert Trajectories tion, our specific formulation of task intention score $1 - \tilde{L}(x_i)$ uses the following combination of losses:

$$1 - \tilde{L}(x_i) = \sum_{j=1}^{3} w_j \cdot (1 - \tilde{L}_j(x_i)),$$

where $\tilde{L}_1, \tilde{L}_2, \tilde{L}_3$ are the normalized symmetry, closure, and smoothness loss, respectively, and the weights ($w_1, w_2, w_3 \in [0,1], \sum_j w_j = 1$) signify the relative priority of each characteristic, allowing for task-specific tuning. It is crucial to note that additional task-specific loss components can be integrated into $\tilde{L}(x_i)$, which may be guided by the user preferences.

3.2 Step 1: Canonical Motion Extraction with Periodic DMP

In this step, periodic DMPs [18] are used to capture a smooth and temporally consistent motion pattern from a single demonstration, rather than denoising the entire dataset. We have the core assumption that within the set of naive demonstrations, there exists at least one trajectory that reflects the best spirit of the task. To implement the shape extraction procedure, we follow the steps below:

1. The expert-like trajectory is selected using the demo utility score $\tilde{U}_i$ (as in (9)), defined over the raw demonstration set E, as $x^{\text{best}}(t) := \underset{x_i \in E}{\arg\max}\, \tilde{U}_i(x_i)$.
2. After learning from the best raw demonstration ($x^{\text{best}}(t)$), the Periodic DMP generates a canonical reference for the desired motion, denoted as $x^d(t)$.
3. Rollouts from the learned Periodic DMP are then used to estimate a state visitation value map $V_E(s) : \mathbb{R}^2 \to [0,1]$ (as shown in Fig. **??**) over the task space, by estimating state visitation density using the nonparametric kernel density estimation (KDE) technique (described in [19]). Formally, for a set of rollout states $\{x_i(t_k)\}_{k=1}^N$ in a 2D task space, the KDE estimate at a point $x_i(t_k)$ is given by$\hat{p}(x_i(t_k)) = \frac{1}{Nh^2} \sum_{i=1}^N K\left(\frac{x - x_i(t_k)}{h}\right)$, where K is a kernel function (e.g., Gaussian) and h is the bandwidth controlling smoothness. The resulting $\hat{p}(x)$ serves as the state visitation value map. This approach is advantageous because it does not require discretizing the task space arbitrarily. This value map guides the reward $R_\pi \in \mathbb{R}$ calculation for the learned policy $\pi(s) : \mathbb{R}^2 \to \mathbb{R}^2$, defining the map from state to action, in the refinement stage, required for adapting the demonstrator's ability factor β during training.
4. To preserve the variability present in the dataset, the shape trajectory $x^d(t)$ is inverse transformed as per the rotation and scaling parameter, derived from the best trajectory $x^{\text{best}}(t)$'s demonstrator preference, in order to obtain the global shape reference $x^g(t)$ with $\theta = 0, \rho = 1$, i.e., no demonstrator bias.
5. For every raw demo $x_i(t)$, the corresponding shape reference is generated from $x^g(t)$, using its respective demonstrator's rotation and scaling parameters θ, ρ, giving $x_i^{\text{aligned}}(t)$.
6. Calculate the loss $L_4(x_i)$ as the distance between the respective demonstrator canonical trajectory and raw demo:

$$L_4(x_i) = \frac{1}{M} \sum_{i=1}^{M} \sum_{k=1}^{N} \left\| x_i^{\text{aligned}}(t_k) - x_i(t_k) \right\|_2. \tag{10}$$

A key result of this stage is the shape preservation score $1 - \tilde{L}_4(x_i)$ (normalized as in (8)), quantifying the accuracy with which the canonical representation reflects the fundamental shape shared among the set of imperfect demonstrations. This loss, along with other intentional loss components

such as (5),(6),(7), will then be used in the subsequent learning based trajectory refinement stage for optimization as

$$1 - \tilde{L}(x_i) = \sum_{j=1}^{4} w_j \cdot (1 - \tilde{L}_j(x_i)). \tag{11}$$

3.3 Step 2: Refinement with Demo utility score

We introduce a Long Short-Term Memory (LSTM) model for refinement and correction. LSTMs are particularly well suited for capturing long-term temporal dependencies [20], enabling the model to learn context-aware corrections essential for complex movements. The LSTM is trained as a sequence-to-sequence model that maps a noisy demonstration trajectory $x_i(t) \in E$ to a denoised and structurally refined output trajectory $\hat{x}_i(t)$. The training objective for the network is to learn this mapping by maximizing the demo utility score $\tilde{U}_i$ defined in (9), using (5), (6), (7), and (10). This formulation enables the LSTM to learn corrections that align the raw input trajectories with the task-intended behavior without requiring explicit supervision through human interventions.

During training, the demonstrator ability factor β is adapted based on the rollout trajectory generated by the LSTM policy $\pi(s)$. Specifically, the reward R_π for a rollout is computed by summing the negative log-likelihood of the state visitation probabilities $V_E(s)$ of all states visited by the trajectory:

$$R_\pi = \sum_{s \in \text{Rollout}(\pi)} V_E(s). \tag{12}$$

The change in the mean reward $\hat{R}_\pi^{(k)}$ per epoch k is then used to update β_i via a $\tanh$ function, for increased sensitivity via $\gamma \in \mathbb{R}$ even with small change in reward:

$$\Delta \hat{R}_\pi^{(k)} = \gamma(\hat{R}_\pi^{(k)} - \hat{R}_\pi^{(k-1)}),$$
$$\beta_i^{(k+1)} = \text{clip}\Big(\beta_i^{(k)} + \tanh(\Delta \bar{R}_k),\ 0, 1\Big),$$

If the reward decreases, β_i is reduced, increasing the emphasis on maximizing mutual information $\mathcal{I}(S; A)$ to guide learning towards the embedded optimal policy. Conversely, if the reward increases, β_i is raised, placing more weight on minimizing the task intention loss to ensure adherence to task objectives. In general, this adaptive principle for tuning β_i allows the LSTM to progressively enhance the behavior of the embedded demonstrator, required to clean the raw demonstrations. Even when the inputs are significantly distorted, the model is able to produce outputs that closely resemble the expert demonstration in shape, periodicity, and symmetry (as conceptually shown in Fig. 1 and illustrated by results in Fig. 2). This two-stage refinement process, leverages the generalization capabilities of DMPs for canonical motion extraction and the expressiveness of LSTMs for fine-grained refinement, which provides a robust framework for learning from a small number of non-expert or degraded demonstrations. The final refined trajectory, denoted as $E_{\text{cleaned}} = \{\hat{x}(t_k)\}_{k=1}^{N}$, where $\hat{x}(t_k)$ represents the refined state at discrete time t_k (conceptually derived from an integral such as $x^*(t) = \int \hat{f}(\cdot)dt$), then can be used as a high-quality target for any subsequent control or tracking systems such as the Neural ODE based low-level controller design as in [21], which is validated with experiments in Section 5.

4 Comparison and discussions

We have raw demonstrations for wiping task (WP), pick-and-place task (PnP) and weaving task (WV), where $N = 300$, $M = 4$, with rotation parameters $\{\theta_i\}_{i=1}^{4} = \{5°, -5°, 10°, -10°\}$ and $\{\rho_i\}_{i=1}^{4} = \{1, 1.1, 1.2, 1.3\}$. These variations in θ and ρ introduce demonstrator preferences, allowing us to evaluate the robustness of our framework in recovering the underlying trajectory shapes from distorted data. We then train multiple imitation learning methods, including Behavioral Cloning (BC) [22], Trajectory-based BC (Traj-BC) [23], Imitation Learning by Estimating Expertise of Demonstrators (ILEED) [6], and Neural ODE (NODE) [21] on both cleaned and noisy datasets.

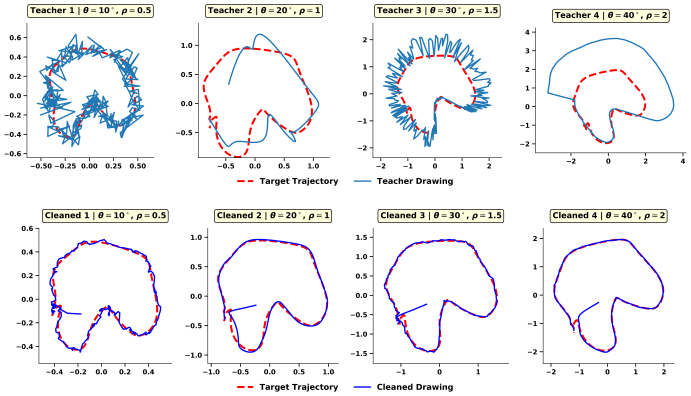


Figure 2: Teacher demonstrations $\{x_i(t_k)\}_{k=1}^4$ shown as solid blue lines in first row, their corresponding shape reference trajectories $\{x_i^r(t_k)\}_{k=1}^4$, denoted as target trajectory, shown as dotted red line, and the resulting cleaned outputs $\{x_i^*(t_k)\}_{k=1}^4$ shown as solid blue lines in second row. Each demonstration is refined using its shape reference to produce a consistent and structured set of cleaned demonstrations.

The resulting outputs, corresponding to 50 rollouts from each trained models as shown in Fig. 3 show that the standard IL methods perform well especially with respect to the loss L_4 for shape preservation with dataset E_{cleaned}, demonstrating the effectiveness of our approach for complex imitation learning tasks like periodic motions.

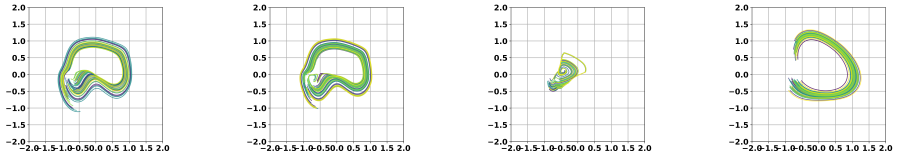
It also validates the statistical advantage by evaluating the performance of the rollout trajectories against the reward R_π (as computed in (12)), which is higher for rollout states with higher value in the map V_E, estimated by the single expert-like trajectory, as shown by the bar chart in Fig. 4. Our method offers the highest statistical advantage in behavior cloning when integrated with ILEED, with increase in mean normalized reward for wiping task by 0.88, pick-and-place task by 0.418, and weaving task by 0.084, as it combines knowledge of demonstrator skill with a corrective refinement mechanism that effectively "skills up" the demonstrator, reducing reliance on knowledge of the complete state space with large dataset. This synergy allows data-hungry ILEED to achieve significantly improved performance even with a small dataset, containing as few as four trajectories, highlighting the robustness and data-efficiency of our approach.

The Neural ODE (NODE) model demonstrates consistently stronger performance across all tasks when trained on the cleaned dataset compared to the noisy dataset. This improvement can be attributed to the higher mean demo utility score of the cleaned dataset, preserving high mutual information between the state and action sequences in the cleaned data, which allows the model to better encode the underlying dynamics. This effect is particularly important in the weaving task, where capturing long-range dependencies and the structure of the two conjoined loops is crucial for accurate trajectory generation.

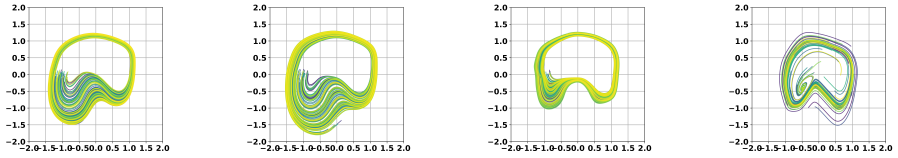
BC and Traj-BC models achieve good performance on the wiping task, benefiting from task intention scores that promote symmetry and smooth closure despite limited data. However, their strong data dependence prevents them from learning the more complex pick-and-place and weaving tasks with only four demonstrations, as evidenced by the high variance in normalized rewards. However, the rollout trajectories for BC and Traj-BC models trained on the cleaned dataset are able to preserve the shape for the required task implementation as compared to the models trained on the noisy dataset, especially in the case of the weaving task.

5 Experimental Results

We performed experiments on the Franka Research 3 Panda robot using hardware setup as shown in Fig. 5. The 2D demonstration trajectories in $E, E_{cleaned}$ are padded with height of 0.3 unit along z-axis and centered about the point $(0.35, 0)$ for 3D trajectories required for wiping task. Similarly,

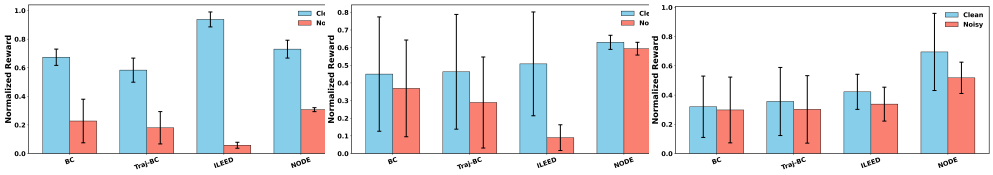


(a) BC (b) Traj-BC (c) ILEED (d) NODE



(e) BC (f) Traj-BC (g) ILEED (h) NODE

Figure 3: Comparison of rollouts across four IL models trained on noisy (top row) and cleaned (bottom row) demonstrations for wiping task.

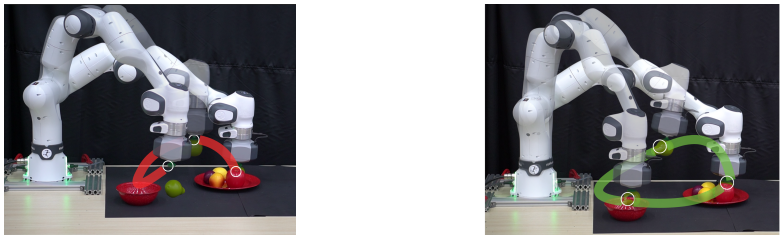


(a) Wiping task (b) Pick-and-place task (c) Weaving task

Figure 4: Comparison of mean reward and reward variability (standard deviation) for rollout trajectories generated by models trained on noisy (in red) versus cleaned (in blue) demonstration datasets.

the demonstration trajectories were aligned in 3D space for the pick-and-place task. In the first scenario, we learned the nominal trajectory using *only 4 naive demonstrations for the wiping task*, with the help of the NODE framework [21]. The resulting motion was inconsistent and failed to preserve the desired specifications as shown in Fig. 5a, 5c. With $E_{cleaned}$, the learned trajectory aligned closely with the intended expert-like drawing. The corrected trajectory retained the essential curvature, periodicity, and closure properties, demonstrating the model's ability to robustly recover trajectories as shown in Fig. 5b, 5d.

(a) Wiping task - noisy (b) Wiping task - clean



(c) Pick-and-place task - noisy (d) Pick-and-place task - clean

Figure 5: Experimental results with expert-like behavior using imperfect demonstrations. Here "noisy" and "clean" means rollout are from IL model trained on dataset E and E_{cleaned}, respectively.

References

[1] B. D. Argall, S. Chernova, M. Veloso, and B. Browning. Robot learning from demonstration: A review. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.

[2] C. Celemin, R. Pérez-Dattari, E. Chisari, G. Franzese, L. de Souza Rosa, R. Prakash, Z. Ajanović, M. Ferraz, A. Valada, and J. Kober. Interactive imitation learning in robotics: A survey. *Foundations and Trends® in Robotics*, 10(1-2):1–197, 2022.

[3] G. Papagiannis and E. Johns. Miles: Making imitation learning easy with self-supervision. In *Proceedings of The 8th Conference on Robot Learning*, pages 810–829, 2025.

[4] H. Yu, Y. Zhu, T. Li, and W. Wu. Co-imitation learning without expert demonstration. *arXiv preprint arXiv:2103.14823*, 2021.

[5] J. Su, Y. Li, Z. Wu, Z. Chen, S. Li, and R. Li. Task-oriented self-imitation learning for robotic autonomous skill acquisition. *International Journal of Humanoid Robotics*, 21(02):2450001, 2024.

[6] M. Beliaev, A. Shih, S. Ermon, D. Sadigh, and R. Pedarsani. Imitation learning by estimating expertise of demonstrators. In *International Conference on Machine Learning*, pages 1732–1748. PMLR, 2022.

[7] Z. Cao and D. Sadigh. Learning from imperfect demonstrations from agents with varying dynamics. *IEEE Robotics and Automation Letters*, 6(3):5231–5238, 2021.

[8] Y. Zhang, Y. Xie, H. Liu, R. Shah, M. Wan, L. Fan, and Y. Zhu. Scizor: A self-supervised approach to data curation for large-scale imitation learning. *arXiv preprint arXiv:2505.22626*, 2025.

[9] S. Belkhale, Y. Cui, and D. Sadigh. Data quality in imitation learning. *Advances in neural information processing systems*, 36:80375–80395, 2023.

[10] M. Du, S. Nair, D. Sadigh, and C. Finn. Behavior retrieval: Few-shot imitation learning by querying unlabeled datasets. *arXiv preprint arXiv:2304.08742*, 2023.

[11] H. Nguyen, A. Baisero, D. Wang, C. Amato, and R. Platt. Leveraging fully observable policies for learning under partial observability. *arXiv preprint arXiv:2211.01991*, 2022.

[12] 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. *arXiv preprint arXiv:2502.08623*, 2025.

[13] S. Haddadin. The franka emika robot: A standard platform in robotics research. *IEEE Robotics & Automation Magazine*, 2024.

[14] K. Eder, C. Harper, and U. Leonards. Towards the safety of human-in-the-loop robotics: Challenges and opportunities for safety assurance of robotic co-workers'. In *The 23rd IEEE International Symposium on Robot and Human Interactive Communication*, pages 660–665. IEEE, 2014.

[15] Z. Yuan, A. W. Hall, S. Zhou, L. Brunke, M. Greeff, J. Panerati, and A. P. Schoellig. Safe-control-gym: A unified benchmark suite for safe learning-based control and reinforcement learning in robotics. *IEEE Robotics and Automation Letters*, 7(4):11142–11149, 2022.

[16] D. Totsila, K. Chatzilygeroudis, V. Modugno, D. Hadjivelichkov, and D. Kanoulas. Sensorimotor learning with stability guarantees via autonomous neural dynamic policies. *IEEE Robotics and Automation Letters*, 2025.

[17] A. Kraskov, H. Stögbauer, and P. Grassberger. Estimating mutual information. *Physical Review E—Statistical, Nonlinear, and Soft Matter Physics*, 69(6):066138, 2004.

[18] A. Ude, A. Gams, T. Asfour, and J. Morimoto. Task-specific generalization of discrete and periodic dynamic movement primitives. *IEEE Transactions on Robotics*, 26(5):800–815, 2010.

[19] B. W. Silverman. *Density estimation for statistics and data analysis*. Routledge, 2018.

[20] Y. Duan, Y. L.V., and F.-Y. Wang. Travel time prediction with lstm neural network. In *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, pages 1053–1058, 2016. doi:10.1109/ITSC.2016.7795686.

[21] F. Nawaz, T. Li, N. Matni, and N. Figueroa. Learning complex motion plans using neural odes with safety and stability guarantees. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 17216–17222. IEEE, 2024.

[22] F. Torabi, G. Warnell, and P. Stone. Behavioral cloning from observation. *arXiv preprint arXiv:1805.01954*, 2018.

[23] C. A. Hepburn and G. Montana. Model-based trajectory stitching for improved behavioural cloning and its applications. *Machine Learning*, 113(2):647–674, 2024.