Compliant Residual DAgger: Improving Real-World Contact-Rich Manipulation with Human Corrections

Xiaomeng Xu* **Yifan Hou*** **Chendong Xin** **Zeyi Liu** **Shuran Song**

Abstract

We address key challenges in Dataset Aggregation (DAgger) for real-world contact-rich manipulation: how to collect informative human correction data and how to effectively update policies with this new data. We introduce Compliant Residual DAgger (CR-DAgger), which contains two novel components: 1) a Compliant Intervention Interface that leverages compliance control, allowing humans to provide gentle, accurate delta action corrections without interrupting the ongoing robot policy execution; and 2) a Compliant Residual Policy formulation that learns from human corrections while incorporating force feedback and force control. Our system significantly enhances performance on precise contact-rich manipulation tasks using minimal correction data, improving base policy success rates by 64% on four challenging tasks (book flipping, belt assembly, cable routing, and gear insertion) while outperforming both retraining-from-scratch and finetuning approaches. Through extensive real-world experiments, we provide practical guidance for implementing effective DAgger in real-world robot learning tasks. Result videos are available at: https://compliant-residual-dagger.github.io/

Keywords

DAgger, Imitation Learning, Manipulation, Compliance Control

1 Introduction

Learning from human demonstrations has seen many recent successes in real-world robotic tasks Chi et al. (2024); Hou et al. (2025); Wu et al. (2025); Xu et al. (2025); Xiong et al. (2025). However, to obtain a successful policy, human demonstrators often have to repeatedly deploy a policy and observe its failure cases, then collect more data to update the policy until it succeeds. This process is broadly referred to as Dataset Aggregation (DAgger) in Ross et al. (2011); Kelly et al. (2019). However, doing DAgger effectively for real-world robotic problems still faces the following challenges:

How to collect informative human correction data? Ross et al. (2011) shows that DAgger is most effective when the correction data is within the original policy's induced state-action distribution. In practice, the common approach is either (1) collecting offline demonstrations that cover the policy's typical failure scenarios Chi et al. (2023), or (2) human taking over robot control during policy deployment Spencer et al. (2020); Mandlekar et al. (2020). However, in both cases, the human demonstrator has no access to the original policy's behavior and may deviate excessively from it. Human taking over additionally introduces force discontinuity when they do not instantly reproduce the exact same robot force. This is partially due to the lack of effective correction interfaces that support precise and instantaneous intervention.

How to effectively update the policy with new data? Prior methods for improving a pretrained policy with additional data include (1) retraining the policy from scratch with the aggregated dataset Kelly et al. (2019), which can be computationally expensive; (2) finetuning the policy with only the additional data Wu et al. (2025); He et al. (2025); Chen et al. (2025), which is sensitive to the quality of the new data Yuan et al. (2024), and (3) training a residual policy separately on top of the pretrained policy, which is typically done with Reinforcement Learning Ankile et al. (2024); Yuan et al. (2024) or Imitation Learning Bharadhwaj et al. (2024), both require a large number of samples.

In this work, we address these questions by proposing an improved system **Compliant Residual DAgger (CR-DAgger)** consisting of two critical components:

- **Compliant Intervention Interface.** We propose an on-policy correction system based on kinesthetic teaching to collect delta action *without interrupting the current robot policy*. Leveraging compliance control, the interface lets humans directly apply force to the robot and feel the magnitude of their instantaneous correction. Unlike take-over corrections, our design allows smooth transition between correction/no correction mode, while providing direct control of correction magnitudes.

- **Compliant Residual Policy.** Leveraging the force feedback from our Compliant Intervention Interface, we propose a residual policy formulation that takes in an *extra force modality* and predicts both residual motions and *target forces*, which can fully describe the human correction behavior. The Compliant Residual Policy is force-aware, even when the base policy is position-only. We show that our residual policy formulation *learns effective correction strategies* using the data collected from our Compliant Intervention Interface.

Together, our system significantly improves the success rate of precise contact-rich robot manipulation tasks using a small amount of additional data. We demonstrate the efficacy of our method on four challenging tasks involving long horizons and sequences of contacts: book flipping, belt assembly, cable routing, and gear insertion. We improve over the base

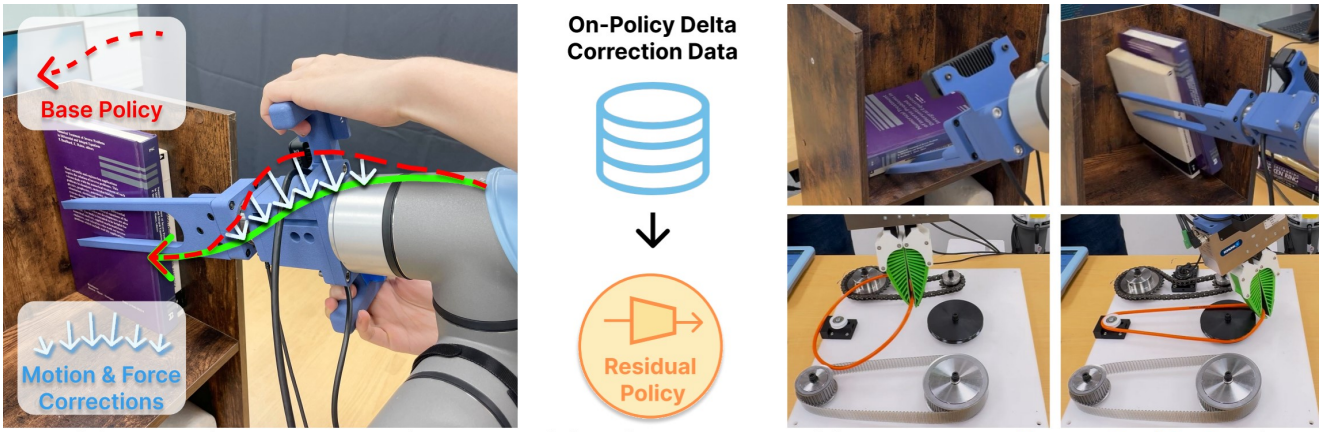

Figure 1. CR-DAgger. To improve a robot manipulation policy, we propose a compliant intervention interface (a) for collecting human correction data, and use this data to update a compliant residual policy (b), and thoroughly study their effects by deploying the updated policy on two contact-rich manipulation tasks in the real world (c).

policy success rate by 64%, while also outperforming retrain-from-scratch and finetuning under the same data budgets. In summary, our contributions are:

- A **Compliant Intervention Interface**, a system that allows humans to provide accurate, gentle, and smooth corrections in both position and force to a running robot policy without interrupting it.
- A **Compliant Residual Policy**, a policy formulation that seamlessly integrates additional force modality inputs and predicts residual motions and forces.
- A **practical guide** for efficient DAgger based on extensive real-world studies for critical but often overlooked design choices, such as batch size and sampling strategy. Our hardware design, training data, and policy code can be found here.

This work is an extended version of the conference paper Xu* et al. (2025). We expand the content of this paper in the following ways:

- Enrich the experimental results with two additional contact-rich tasks as detailed in Sec. 4: a peg-in-hole style gear insertion task, which requires sub millimeter accuracy and tests our system for small-magnitude correction motions; and a cable routing task, which contains large variations in item physical properties (cable stiffness).
- Identify and explain the **intention-misinterpretation** phenomenon in correction data collection in Sec. 3.1, caused by robot tracking errors. We provide a solution to the problem by reducing tracking error both spatially and temporally and include additional analysis on the effect of tracking error in Sec. 4.5.
- Analyze the quality improvement of human correction data from our proposed system comparing with traditional manipulation data collection method in Sec. 4.4.

2 Related Work

Human-in-the-Loop Corrections for Robot Policy Learning. The original DAgger work Ross et al. (2011) requires the demonstrator to directly label actions generated by the policy. In robotics, a practical variation is to let the human *take over* the robot control and provide correct action directly Kelly et al. (2019). Such human correction motion can be recorded with spacemouse Chen et al. (2025); Liu et al. (2022), joystick Spencer et al. (2020), smartphone Mandlekar et al. (2020), or arm-based teleoperation system Hoque et al. (2021a,b); Wu et al. (2025). We instead proposes a novel kinesthetic teaching system with compliance controller that allows the demonstrator to apply delta corrections while the robot policy is still running, and additionally records force feedback. Our results show that both the delta correction data and the force data are crucial to the success of the learned policy.

Improving Pretrained Robot Policies with New Data. The most direct approach to improving a pretrained policy with new correction data is to retrain the policy on the aggregated dataset, combining prior demonstrations with new feedback Mandlekar et al. (2020); Spencer et al. (2020). Alternatively, Reinforcement Learning (RL) offers a framework to incorporate both offline and online data, either by warm-starting replay buffers Luo et al. (2024); Ball et al. (2023) or by using offline data to guide online fine-tuning Yin et al. (2025); Xu et al. (2022). When policies are trained on large-scale human demonstration datasets Black et al. (2024); Team et al. (2024); Kim et al. (2024); O'Neill et al. (2024); Khazatsky et al. (2024); Xu et al. (2023a, 2024a), retraining becomes impractical, especially when the original data is inaccessible. In such cases, fine-tuning with only the new data is a common solution, using either imitation learning Liu et al. (2022); He et al. (2025); Wu et al. (2025) or RL Mark et al. (2024); Chen et al. (2025). Another line of work introduces an additional residual model on top of the original policy. These residual policies can be trained with RL in simulation Yuan et al. (2024); Ankile et al. (2024); Haldar et al. (2023a), but suffers from sim-to-real challenges. Training residual policy in the real world usually requires a large number of samples Bharadhwaj et al. (2024); Johannink et al. (2019), intermediate scene representation Guzey et al. (2024b), or consistent visual observations between training and testing Guzey et al. (2024a); Haldar et al. (2023b), making

the approach hard to adopt in practice. In this work, we introduce a practical data collection system and an efficient residual policy learning algorithm for long-horizon, contact-rich manipulation tasks. Our approach requires only a small amount of real-world correction data and supports integration of additional sensory modalities not present in the original model, leading to improved policy performance.

Compliance and Compliance Control. Compliance is a motor property that describes how motion responds to force. For example, traditional industrial robots typically have position control with very low compliance to achieve precise tracking under heavy load. On the contrary, high compliance lets a robot retreat when it experiences external forces. A high compliance or a variable compliance profile Mason (2007) may be preferred for tasks involving interactions with the environment, since they can increase execution robustness or avoid huge contact forces during the interactions Hou and Mason (2019); Hou et al. (2020).

Compliance control refers to feedback control techniques that give a robot high compliance or spatial-temporally varying compliance profiles. Compliance control can be implemented by Impedance Control Hogan (1984) if the robot is back-drivable, or by Admittance Control with external force sensors if the robot cannot be back-driven, such as most industrial robots. A more through review of compliance controller implementations can be found in Lynch and Park (2017) and Villani and De Schutter (2016).

3 CR-DAgger Method

Our goal is to improve a pretrained robot policy with a small amount of human correction data. To achieve this, we propose a Compliant Intervention Interface (§ 3.1) that enables precise and intuitive on-policy human correction data collection, and a Compliant Residual Policy (§ 3.2) that efficiently learns the correction behaviors to be used on top of the pretrained policy. Throughout the paper, we use the term *base policy* to refer to the pretrained policy without online improvements.

3.1 Compliant Intervention Interface

Correction data is collected by human demonstrators to rectify policy failures. Unlike initial demonstrations that establish baseline behaviors, correction data specifically targets failure modes observed during policy deployment. Correction data is most effective when it corrects failures in policy-induced state distributions Ross et al. (2011). The interface through which these corrections are collected significantly impacts the quality of correction data, which should be intuitive for demonstrators, capture critical corrective information at precise moments of failures, and facilitates collecting data close to the base policy state-action distribution.

There are two types of correction collection methods: *Off-policy correction* is when humans observe failures of the base policy during deployment, and then recollect extra offline demonstrations to address failure cases. This approach is most commonly used for improving Behavior Cloning policy performance due to its simplicity - it requires *no additional infrastructure* beyond the original data collection setup. However, the resulting demonstrations may fail to cover all the failure cases or deviate from the original state-action distribution. We focus on *on-policy correction* instead, where humans can monitor policy execution and intervene on the spot when failures occur or are anticipated. This approach allows humans to provide corrections more targeted to the base policy's failure cases. However, challenges still exist for an intervention system:

- **Non-smooth transitions**. Intervention in robotics is typically implemented by *take-over* correction: letting human take complete control and overwrite robot policy. As the underlying control abruptly switches between robot policy and human intention, disturbances are introduced due to policy inference and human response latency, especially when the robot is withholding external forces. The recorded data thus may include undesired actions that do not reflect the human's intention.

- **Distribution shift.** The human-intervened state-action distribution may deviate significantly from the original distribution. Additionally, the non-smooth transition above could bring in disturbances and add to the distribution shift.

- **Indirect correction brings errors.** Correction is commonly implemented via teleoperation interfaces such as spacemouse or joysticks He et al. (2025); Chen et al. (2025). With spatial mismatch and teleoperation latency, it is hard for the demonstrator to instantly provide accurate corrections upon intervention starts without going through a short adjustment period.

- **Missing information.** The recorded correction data need to fully describe the human's intended action. Simply recording the robot's position is not sufficient, since it may be under the influence of human correction force and will cause different result when testing without human.

We propose a *Compliant Intervention Interface* with the following designs to solve those challenges:

- **Delta correction instead of take-over correction.** Unlike take-over correction, where the demonstrator has no idea of the policy's original intention once taking over, we propose a novel on-policy delta correction method: we let the robot policy executes continuously while the human applies forces to the robot with a handle mounted on the end effector, resulting in delta actions on top of the policy action. The human demonstrator can always sense the policy's intention through haptic feedback, and easily control the magnitude of intervention by the amount of force applied to the handle. As a result, delta correction ensures smooth intervention data and limits the human from providing very large corrections. The approach is also intuitive as the human can directly move the robot towards desired correction directions.

- **Correction interface with compliance control.** In order to apply delta correction over a running policy, we provide a compliant interface that allows humans to safely intervene and apply force to the robot to affect its behaviors at any time, as shown in Fig. 2. We design a kinesthetic correction hardware setup with a detachable handle for human to hold when correcting, and allows easy tool-swapping for different tasks. We run a compliance controller (specifically admittance control) in the background to respond to both

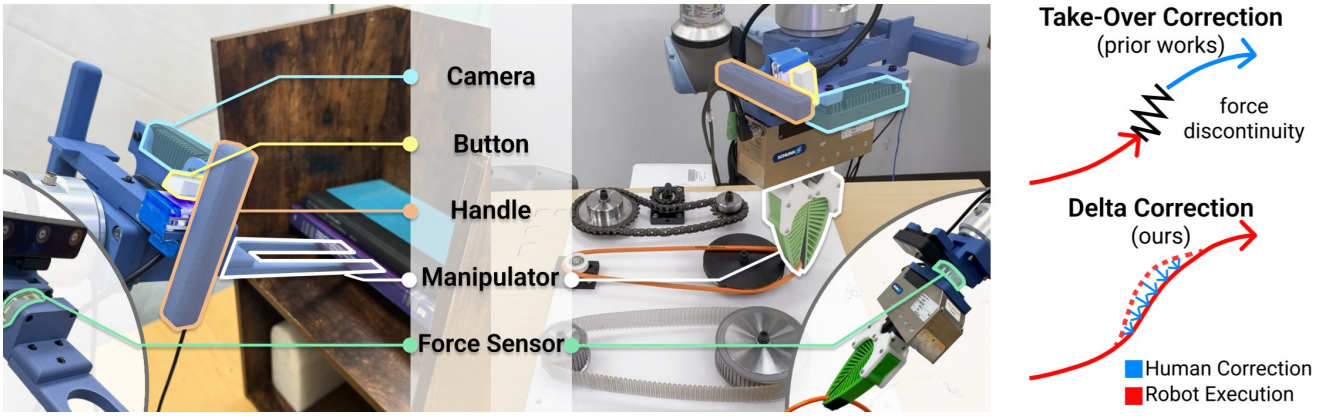

Figure 2. Compliant Intervention Interface characterized by a kinesthetic correction hardware setup where humans hold on the handle and apply forces to correct robot execution, providing on-policy delta corrections.

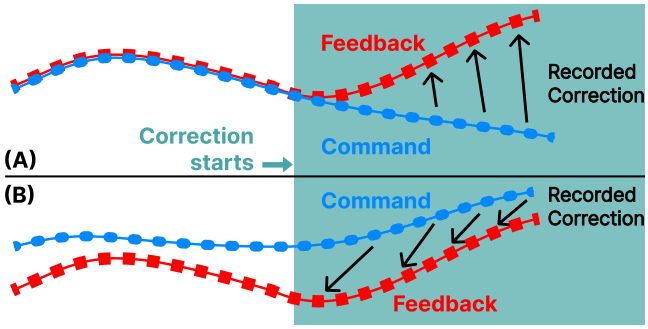

Figure 3. Illustration of the intention misinterpretation caused by tracking error. In both cases, the robot feedback (and thus the human correction motion) are exactly the same. However, the recorded correction motion direction is very different. (A) recorded correction is correct when the tracking error is small. (B) recorded correction motion has wrong direction when there is large tracking error caused by both control and latency.

contact forces and human correction forces, allowing the human to influence but not completely override the policy execution. The admittance controller forms a virtual spring-mass-damper system around the base policy output q_{ref}:

$$M\ddot{q} = K(q_{ref} - q) - D\dot{q} + F, \tag{1}$$

where M, K and D denotes the virtual inertia, stiffness and damping of the compliant system, F represents the combined force feedback from both external contacts and human correction. The admittance controller uses a constant stiffness $\sim$1000 N/m to allow easy human intervention and ensure accurate tracking.

- **Correction recording with buttons and force sensor.** Our interface additionally includes an ATI 6-D force sensor to directly measure contact forces, and a single-key keyboard placed on the handle to record the exact timings of correction starts/ends. Both the policy's original commands and the human's delta corrections are recorded, along with force sensor readings during the interaction.

Additionally, comparing with the conference version of this paper Xu* et al. (2025), we identified and fixed an issue of **intention misinterpretation**: When the human correction motion is smaller than the robot tracking error, the human intention can be misinterpreted and the residual policy would learn the wrong correction motion. This is illustrated in Fig. 3. With our Compliant Intervention Interface, human correction motion is computed as the difference between robot position feedback and the base policy output. A subtle assumption here is that the robot position before intervention accurately reflects the base policy behavior, i.e. the tracking error should be negligible. When the tracking error is larger than the delta corrective motion from human intervention, the computed correction motion could be in the opposite direction of the human's true intended correction direction. To avoid intention misinterpretation even when human correction is small, we implement two features on top of Xu* et al. (2025) to reduce the tracking error: 1) we add a velocity tracking term in the compliance control law in addition to the position tracking term used in 1:

$$M\ddot{q} = K(q_{ref} - q) + D(\dot{q}_{ref} - \dot{q}) + F, \tag{2}$$

where the velocity reference $\dot{q}_{ref}$ is computed via finite difference from the position reference q_{ref}. This reduces the tracking error spatially. 2) We added a look-ahead time to robot feedback when computing the correction motion q_{delta}:

$$q_{delta}[t] = q[t] - q_{ref}[t - \Delta t], \tag{3}$$

where Δt is empirically set by the estimation of hardware latency. The look-ahead time reduces the tracking error temporally. Together, the largest tracking error reduces from 30 mm to below 5 mm during fastest robot motion in our experiments.

3.2 Compliant Residual Policy

Given the correction data, there are multiple ways to update the policy. Common practices include *retraining* the base policy from scratch with both initial data and correction data, and *finetuning* the base policy with only the correction data. However, *retraining* is costly as it requires updating the entire base policy network from scratch with all the available data. It also requires access to the base policy's initial training data, which might not be accessible for many open source pretrained models. The amount of correction data is significantly smaller than the initial training data, thus simply mixing them together makes the policy hard to gain effective corrective behaviors. While *finetuning* allows updating partial policy network parameters with new data

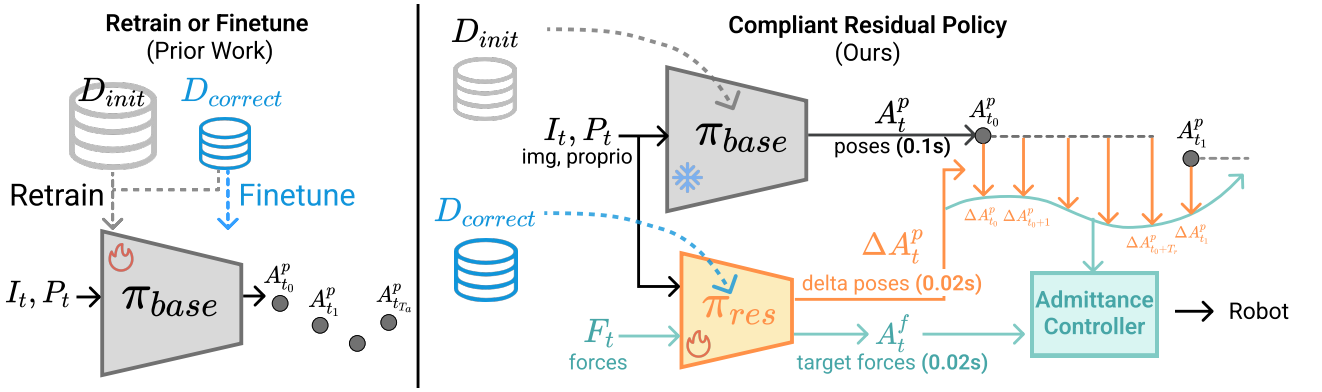

Figure 4. Policy Update Methods. Left: Common policy update methods - retraining and finetuning. Right: Ours. The base policy runs at 1 Hz. It takes in images I_t and proprioceptions P_t and predicts 32 frames of end-effector poses $A^p_t = \{A^p_{t_0}, A^p_{t_1}, ..., A^p_{t_{T_a}}\}$ spaced 0.1 Seconds apart. The Compliant Residual Policy runs at 50 Hz. It takes in additional force inputs F_t and predicts 5 frames of delta poses $\Delta A^p_t = \{\Delta A^p_{t_0}, \Delta A^p_{t_0+1}, ..., \Delta A^p_{t_0+T_r}\}$ and target forces A^f_t spaced 0.02 Seconds apart. The combined poses of A^p_t and ΔA^p_t, and target forces A^f_t are taken by an admittance controller to command the robot.

only, its training stability can be easily affected by the distribution mismatch between the correction data and initial training data. Moreover, both retraining and finetuning can only update the policy with its fixed network architecture while being unable to incorporate new inputs and outputs. We propose a compliant residual policy trained only on the correction data to refine base policy's position actions and predict additional force actions.

Compliant residual policy formulation. Our policy directly learns corrective behavior from the human delta correction data, as shown in Fig. 4. It takes as input the same visual and proprioceptive feedback as the base policy but with a shorter horizon. It also takes in an extra force modality, which is available using our Compliant Intervention Interface. The policy outputs five frames of actions at a time, corresponding to 0.1 s of execution time when running at 50 Hz. The action space is 15-dimensional: the first nine dimensions represent the SE3 delta pose from the base policy action to the robot pose command Chi et al. (2023), while the later six dimensions represent the expected wrench (force and torque) the robot should feel from external contacts. Both the robot pose command and the expected wrench are sent to a standard admittance controller for execution with compliance.

The residual policy directly uses the base policy's frozen image encoder Radford et al. (2021); Amir et al. (2021); Xu et al. (2023b) to extract an image embedding, a temporal convolution network Van Den Oord et al. (2016) to encode the force vectors, followed by fully-connected layers to decode actions.

Advantages. This formulation provides the following advantages:

- *Sample-efficient learning.* The residual policy's network is light-weight (∼2MB trainable weights) and only requires a small amount of correction data to train (50∼100 demonstrations).

- *Incorporating new sensor modality.* Compared to retraining and finetuning methods that are limited to the base policy's network architecture, residual policy can incorporate new sensor modality. This allows taking any position-based pretrained policy and turning it force-aware simply by collecting a small amount of correction data with force modality.

- *High-frequency inference.* The light-weight residual policy runs at a higher frequency than the base policy, incorporating high-frequency force feedback and enabling reactive corrective behaviors. This reactivity is particular important for error correction during contact events.

Training strategy. In prior work, a residual policy is trained either in simulation with RL Ankile et al. (2024); Yuan et al. (2024) to give it sufficient coverage of the state distribution, or in the real world with pre-collected behavior cloning data Luo et al. (2024). In this work, we train the Compliant Residual Policy completely on the small amount of new real-world correction data with the following strategies:

- *Ensure sufficient coverage of in-distribution data.* Human correction tends to be frequent around a few key moments of the task. A residual trained on correction data alone can extrapolate badly around states where no correction is provided. To help the residual policy understand when *not* to provide corrections, we: (1) include the no correction data for training but label it as zero residual actions; (2) collect a few trajectories where the demonstrator always holds the handle and marks the whole trajectory as correction even when the correction is small or zero. Details are in § A.3.

- *Prioritize correction data over no-correction (zero residual action) data.* Similar to Liu et al. (2022), we alter the data sample frequency during training based on whether they have human correction or not. Specifically, since the moment of correction start indicates where the current policy performs badly followed by immediate action to fix it, we sample data more frequently for a short period immediately after correction starts. Our real-world ablations (§ 4.5) demonstrate that our training strategies improve the quality of the residual policy and the overall success rate.

4 Evaluation

For each task, we train a diffusion policy Chi et al. (2023) with 150∼400 demonstrations as the base policy. We first

deploy the base policy and observe its performance and failure modes. Next, from the same base policy, we collect 50∼100 correction episodes with each data collection method. Then, we update the policy using each network updating method and training procedure. Finally, we deploy the updated policies and evaluate their performance under the same test cases. Details of tasks and comparisons are described below.

4.1 Contact-Rich Manipulation Tasks

Book Flipping: As shown in Fig. 5 (a), this task is to flip up books on a shelf. Starting with one or more books lying flat on the shelf, the robot first insert fingers below the book, then rotate the book up and push them firmly against the shelf to let them stand on their own. The base policy is trained with 150 demonstrations.

This task is challenging as it involves rich use of physical contacts and forceful strategies Hou et al. (2020). A position-only strategy always fails immediately by triggering large forces, so we execute all policies through the same admittance controller. The task success requires high precision in both motion and force to accurately align the fingers with the gap upon insertion, and to provide enough force to rotate heavy books and make the books stand firmly.

Each evaluation includes 20 rollouts on 4 test cases (5 rollouts each), as shown in Fig. 5 (b): 1) flipping a single book (three seen and two unseen books), initially far from the shelf edge; 2) flipping a single book close to the shelf edge; 3) flipping two books together (combinations of three seen and three unseen books), initially far from the shelf edge; 4) flipping two books close to the shelf edge. We use the same initial configurations for all evaluations.

Belt Assembly: As shown in Fig. 6 (a), this task is to assemble a thin belt onto two pulleys, which is part of the NIST board assembly challenge Kimble et al. (2020). Starting with the belt grasped by the gripper, the robot needs to first thread the belt over the small pulley, next move down while stretching the belt to thread its other side on the big pulley, then rotate 180° around the big pulley to tighten the belt, and finally pull up to release the belt from the gripper. The base policy is trained with 405 demonstrations.

The task is challenging as it requires both position and force accuracy throughout the process. Specifically, the belt is thin and soft so the initial alignments onto the pulleys are visually ambiguous. Also, since the belt is not stretchable, there is more resistance force and less position tolerance as the belt approaches the second pulley, requiring a policy with good force-position coordination and adaptation. We ran 32 rollouts across four different initial board positions and four grasp locations (Fig. 6 (b)).

Cable Routing The task includes routing a USB cable around three clips, as shown in Fig. 7 (a). The USB cable has one fixed end and a free end. The cable starts being grasped, and can slide easily through a groove on the finger as the hand moves. The base policy is trained with 200 demonstrations.

We make this task challenging by using cables with different appearances and stiffness than those seen by the base policy. A change in cable position or physical property can easily cause the cable to be too high and miss a clip (Fig. 8 (b)), or too low and get stuck by a clip. We additionally introduce difficulty by using clips that are visually different from the one used in base policy training, testing our system's generalization capability. We ran 20 rollouts across five cables and four different fixed end locations.

Gear Insertion The task involves inserting a plastic gear onto a metal axis while mating it to a nearby gear, which is also a part of the NIST board assembly challenge Kimble et al. (2020). The base policy is trained with 150 demonstrations.

The task is challenging due to the tiny tolerance between the gear and the axis. It is also different than the previous tasks as the human correction motions often have small magnitudes. We introduce additional difficulties by elevating the base board by 3 cm. We ran 20 rollouts across five different base locations.

4.2 Base Policy and its Failure Modes

We trained a diffusion policy Chi et al. (2023) that takes in past images from a wrist-mounted camera and robot proprioception observations, and predicts a future position-based action trajectory. To isolate the contribution of force inputs versus human corrections, we trained diffusion policies with and without force inputs as baselines for the belt assembly task.

The book flipping base policy achieves a 40% success rate with the following common failure cases (Fig. 5 (c)): (1) Missed insertion. The fingers initially go too high above the book or aims for the gap between the two books, failing to properly insert beneath the books. (2) Incomplete flipping. At the last stage, the policy retracts the blade before the book can stand stably, causing it to fall back.

The belt assembly base policy achieves a 15.6% success rate. Adding force input increases the base policy success rate to 43.8%. Common failure cases include (Fig. 6 (c)): (1) Missed slotting: the fingertip goes too high or too low, causing the belt to miss the slot on the big pulley. (2) Belt slippage: the fingers pull the belt in the wrong direction, causing the belt to tilt and slip off the pulley.

4.3 Baselines

We compare CR-DAgger with baselines across two dimensions: correction method and policy update method. We present the quantitative results in Fig. 9, and explain key findings in § 4.4.

Correction data collection methods. We compare our Compliant Intervention Interface with the two most commonly used correction data collection strategies:

- *Observe-then-Collect* includes two steps: first, the policy is deployed and human demonstrators observe the initial settings that could cause failures; then, demonstrators provide completely new demonstrations starting from similar initial settings. As explained in § 3.1, this type of offline correction potentially misses critical timing information, and the resulting demonstrations may deviate from the policy's original behavior distribution.

- *Take-over-Correction* (HG-DAgger) Kelly et al. (2019) is another common correction strategy where human demonstrators monitor policy execution and take complete control when failures are anticipated. We implement Take-over-Correction on our Compliant Intervention Interface by cleaning up command buffer to the compliance controller and switching stiffness to zero upon correction starts, so the

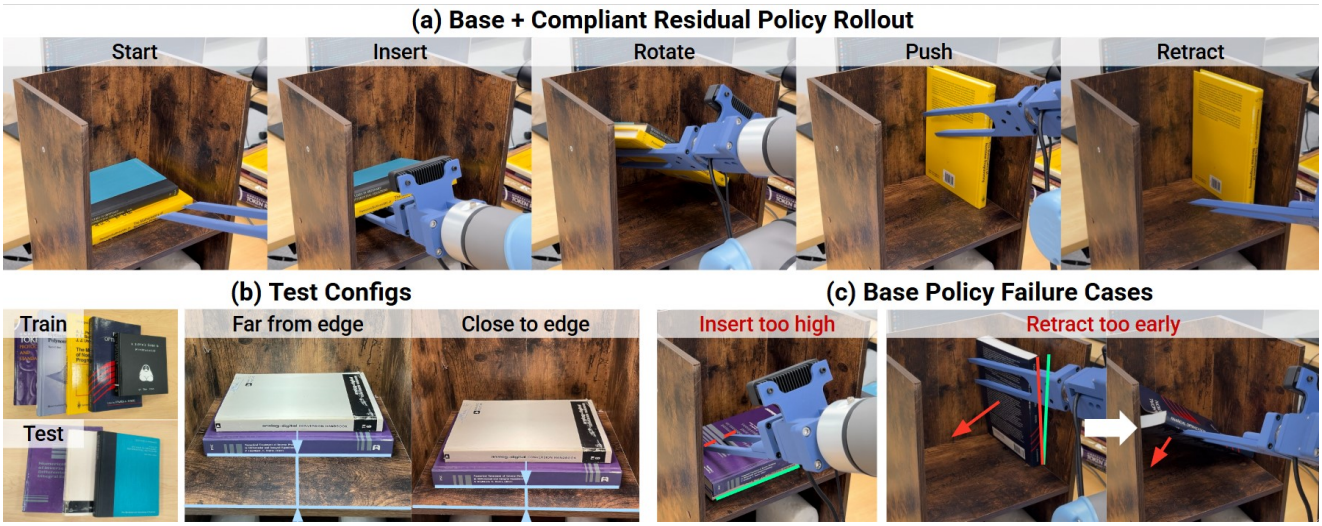

Figure 5. Book Flipping Task. (a) Policy rollout of [Compliant Residual] policy trained with [On-Policy Delta] data, demonstrating accurate insertion motions and forceful pushing strategy. (b) Different test scenarios. (c) Typical failure cases of the base policy: inserting too high above the book and missing the gap; retracting the fingers before the books can steadily stand.

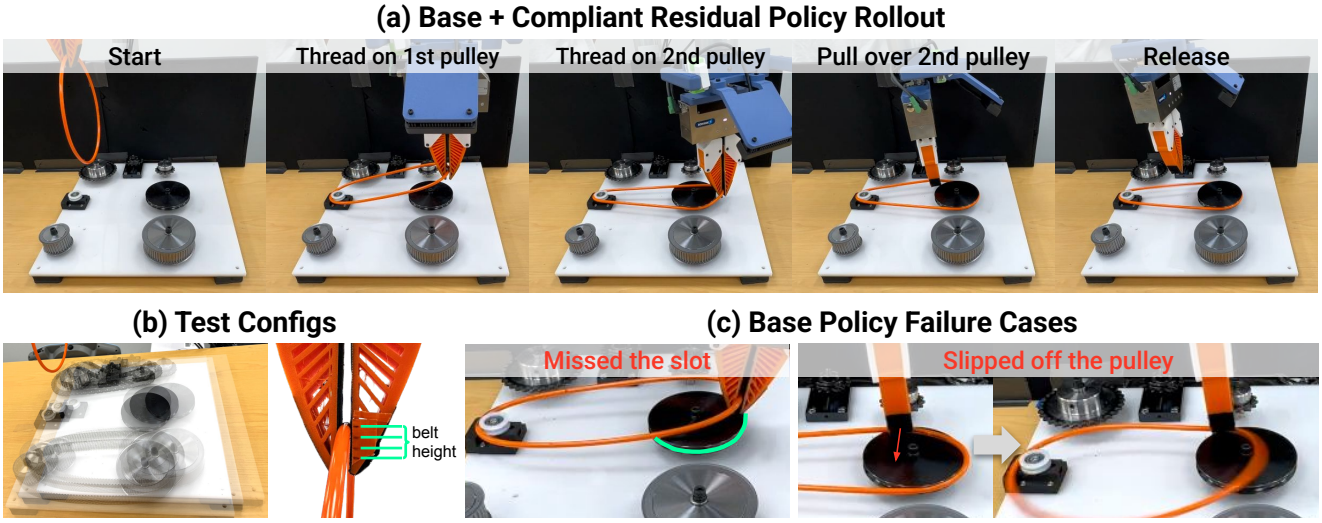

Figure 6. Belt Assembly Task. (a) Policy rollout of [Compliant Residual] policy trained with [On-Policy Delta] data, demonstrating accurate force-position coordination and adaptation. (b) Different test scenarios. (c) Typical failure cases of the base policy: missing the slot by going too high above the pulley; tilting the belt and causing it to slip off the pulley.

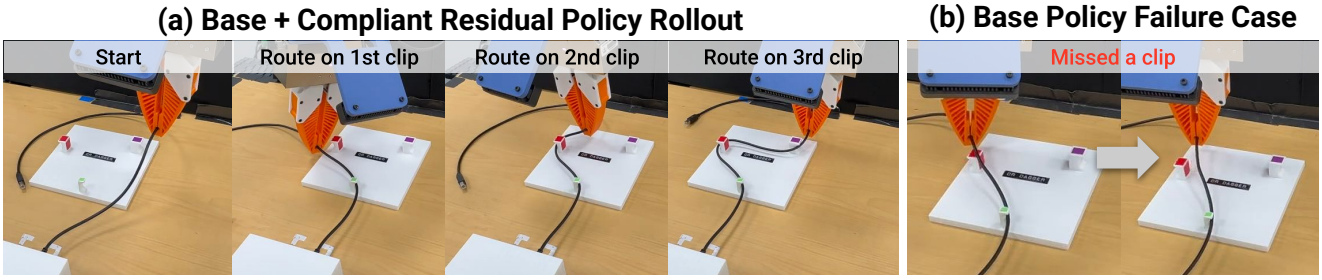

Figure 7. Cable Routing Task. (a) Policy rollout of [Compliant Residual] policy trained with [On-Policy Delta] data. (b) Typical failure case of the base policy: missing a clip.

robot policy does not affect the robot during correction. However, as explained in § 3.1, take-over correction introduces an abrupt transition around control authority switching, which may cause distributional discontinuities in the training data.

- *On-Policy Delta (Ours):* the details are described in § 3.1.

Policy update methods. We compare with two common policy update methods:

- *Retrain Policy:* Retrain the base policy using both the original training data and the correction data from scratch. As explained in § 3.2, this approach is reliable but may require access to the orignal data and large amount of new data to work well.

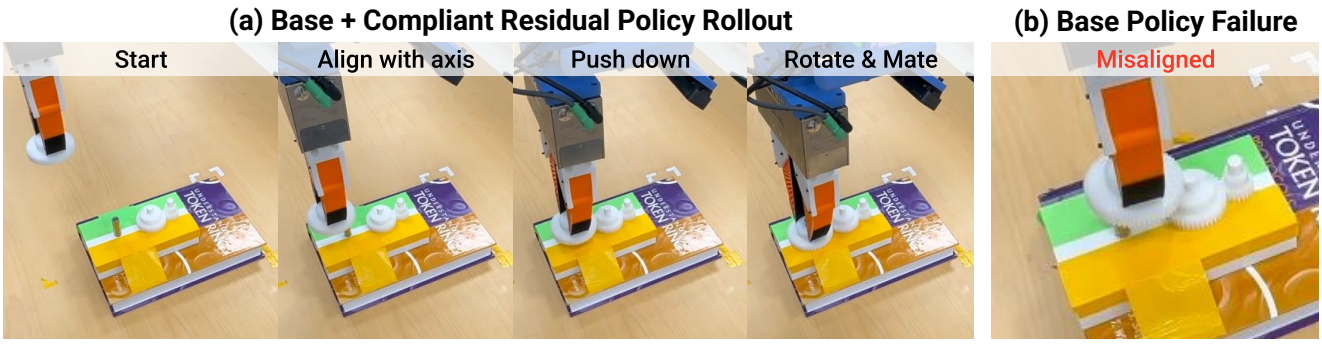

Figure 8. Gear Insertion Task. (a) Policy rollout of [Compliant Residual] policy trained with [On-Policy Delta] data. (b) Typical failure case of the base policy: misaligning the gear with the axis and failing to mate with the nearby gear.

- *Finetune Policy:* Finetune the base policy using only the correction data (freezing visual encoders). As explained in § 3.2, this approach can be sensitive to data quality and distribution shifts.
- *Fintune Policy with KL Regularization:* A recent method Fan et al. (2023) that stabilizes finetuning training by encouraging the predicted action to be close to the training data distribution.
- *Residual Policy:* an ablation of our method where force is removed from both input and outputs.
- *Compliant Residual Policy (Ours):* Residual policy update with additional force input and outputs, see details in § 3.2.

4.4 Key Findings from Comparisons

Finding 1: Compliant Residual Policy is able to improve base policy by a large margin. As shown in Fig. 9, [Compliant Residual] policy trained with [On-Policy Delta] data improves the base policy success rate by 60% and 50% on the two tasks respectively. It effectively learns useful corrective strategies from the limited demonstrations. For example, in the book flipping task, the policy learns to pitch the fingers down more before finger insertion to increase the insertion success; in the belt assembly task, the policy learns to correct the height of the belt when misaligned to the pulley slot. Results are best viewed in our supplementary video.

Finding 2: Residual policy allows additional useful modality to be added during correction. [Compliant Residual] policy performs significantly better than other methods without force (45% higher success rate than the best position-only baseline on the book task and 53% higher on the belt task) as it can both take in force feedback that indicates critical task information and predict adequate contact forces to apply. For example, the last stage of the book flipping task requires the robot to firmly push the book against the shelf wall to let it stand on its own. [Compliant Residual] policy predicts large pushing forces at this stage to make the books stand stably with a 100% success rate, while [Residual]'s success rate drops from 70% to 35% (§ A.2). The second stage of the belt assembly task (threading the belt on the large pulley) requires delicate belt height adjustments under ambiguous visual information due to occlusions and the lack of depth. [Compliant Residual] policy learns to move along the pulley to find the slot when the finger touches the top of the pulley.

Finding 3: Smooth On-Policy Delta data enables stable residual policy. [Compliant Residual] policy has 45% higher success rate when trained on [On-Policy Delta] data instead of [Take-over-Correction] data on the book flipping task. Residual policy trained with [Take-over-Correction] data sometimes exhibits large noisy motions that trigger task failures, such as retracting the fingers too early in the book flipping task. On the contrary, the residual policy trained with [On-policy Delta] data has much smoother action trajectories and better reflects human's correction intentions, providing suitable magnitudes of corrections. To see the improvements more clearly, we plot robot velocity data around correction starts and ends from 100 episodes of data collections in Fig. 10. [Take-over-Correction] data contains larger robot velocity magnitude right after control authority switches, especially when robot takes over control from human. Moreover, the demonstrator can directly perceive the base policy's intention and the extent of correction being applied through the resistance force when employing the [On-Policy Delta] kinesthetic teaching style correction, thus preventing correction demonstrations from deviating too much from the base policy's state-action distribution. Fig. 11 shows that [On-Policy Delta] introduces less distribution shift compared to [Take-over-Correction].

Finding 4: Retraining base policy is stable but learns correction behavior slowly. Retraining from scratch with the initial and correction data together leads to policies with stable motions. However, its behavior is less affected by the small amount of correction data compared to the dominant portion of initial data, leading to insignificant improvements over the base policy (1.67% success rate drop on the book task averaged across all data collection methods, 18.8% success rate improve on the belt task, both are much less improvements than [Compliant Residual]).

Finding 5: Finetuning base policy is unstable. Policy finetuning with either correction data has the worst performance across all policy update methods and even underperforms the base policy (30% success rate drop on the book task averaged across all data collection methods, 15.6% drop on the belt task). The finetuned policy predicts unstable and noisy motions, quickly leading to out-of-distribution states, such as inserting too high in the book flipping task and drifting away from the board in the belt assembly task. This is likely due to the distribution mismatch between the

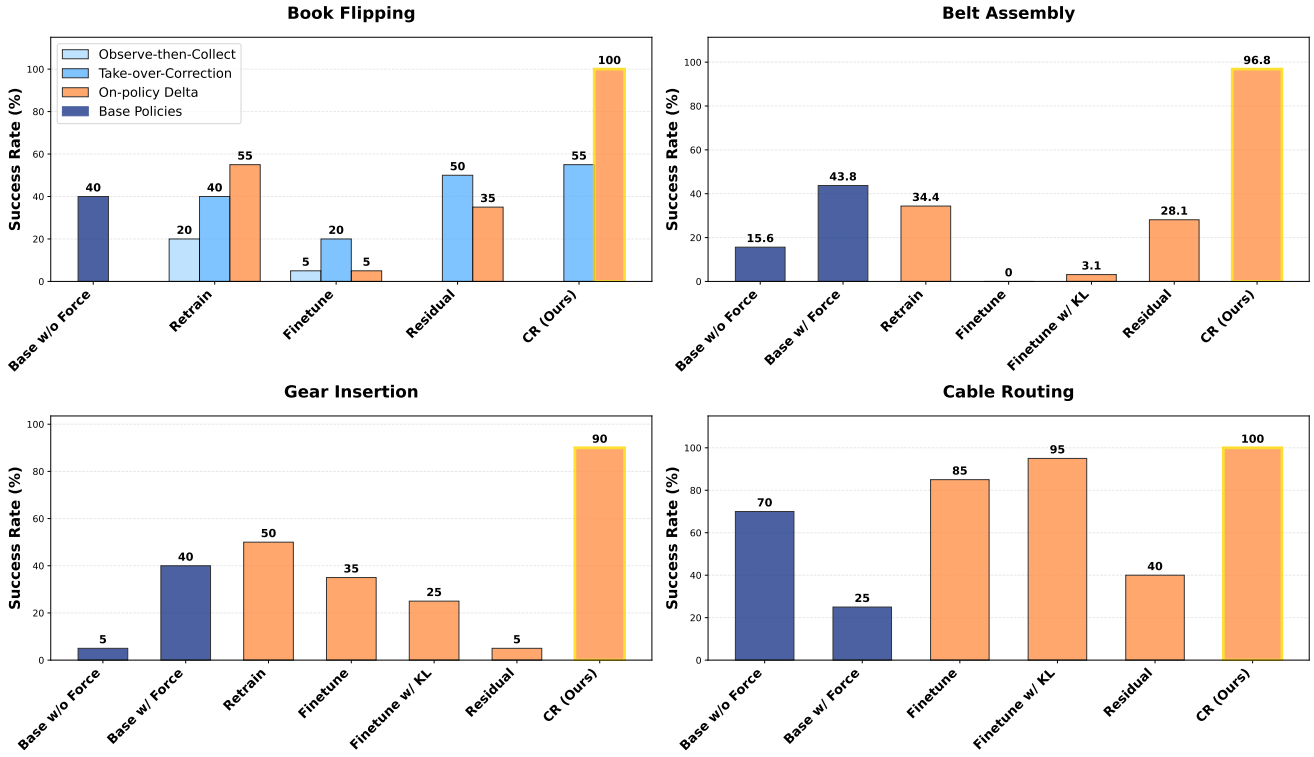

Figure 9. Results. We compare CR-DAgger across two dimensions: correction method and policy update method. The result shows that our [Compliant Residual (CR)] policy trained with [On-Policy Delta] data is able to improve upon base policies on both tasks and outperforms other variations.

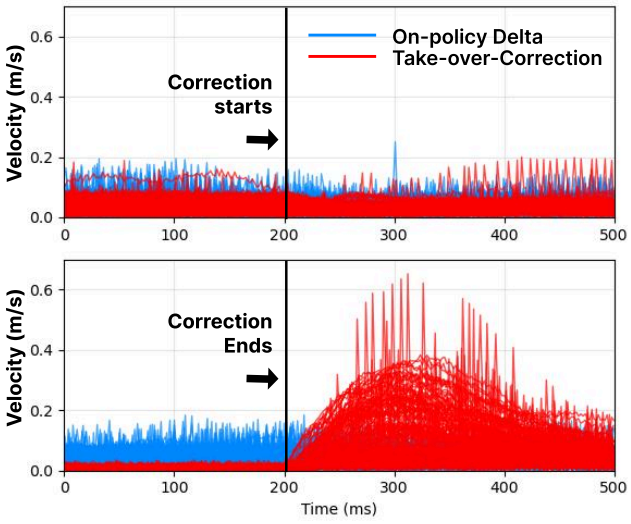

Figure 10. [On-Policy Delta] enables smoother trajectories. Here compares velocity magnitude within 1.5 s of the corrections starts/ends. [On-Policy Delta] velocity magnitudes are smaller and more consistent, [Take-Over Correction] has notably larger magnitude and variations, demonstrating that [On-Policy Delta] encourages smoother trajectories.

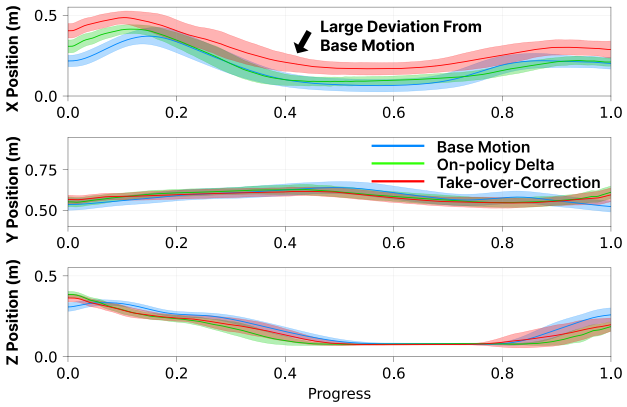

Figure 11. [On-Policy Delta] introduces less distribution shift. Here we compare the distribution of fingertip trajectories among 1) base policy training data, 2) correction episodes from [On-Policy Delta], and 3) correction episodes from [Take-Over-Correction]. For intuitive visualization, all trajectories are stretched to the same duration and normalized to (0,1). The range of the vertical axis is 0.55m for all three axes. [On-Policy Delta] data's distribution is better aligned with base policy training data's distribution than [Take-Over-Correction] data.

base policy training data and correction data, causing training instabilities. Adding KL-regularization effectively reduced the noisy behavior, however, the overall success rate is still lower than other baselines.

4.5 Ablations

We study the effect of important design decisions in our method with ablation studies.

Training frequency and batch size. One important parameter in DAgger is the batch size between policy updates. With a smaller batch size, the policy is updated more frequently, then new correction data can better reflect the most recent policy behavior. However, DAgger with small batch sizes is known to suffer from *catastrophic forgetting* Kirkpatrick et al. (2017); Goodfellow et al. (2013) since it finetunes neural networks on data with non-stationary distribution. Common solutions include retraining the residual

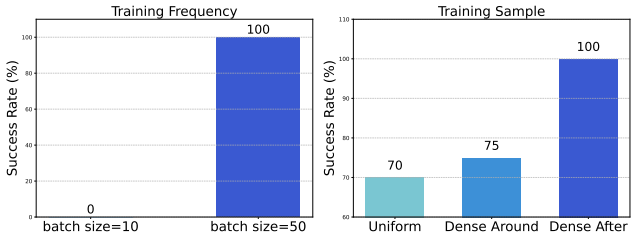

Figure 12. Effect of Training Frequency and Sample. Batch size=50 leads to more stable training and dense sampling after correction starts achieves better performance on the book flipping task.

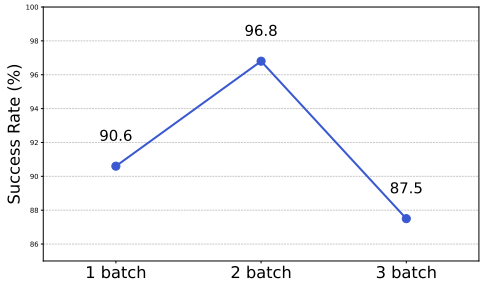

Figure 13. Multi-Batch Update Result. The compliant residual policy trained on two batches of correction data achieves the best result on the belt assembly task, indicating that more batches of correction data do not guarantee better performance.

policy at the end of DAgger using all available correction data collected from all the intermediate residual policies Wu et al. (2025). Another way is to rely on the base policy training data as a normalizer He et al. (2025). In this work, we empirically found that larger batch sizes can effectively stabilize residual training. With batch size = 50, the book flipping task reach 100% with one batch, while the belt assembly task performs better with two batches. We compare our method with a smaller batch size on the book flipping task, where we warm up the residual with 20 episodes of initial correction data, then update every ten more episodes for three times. We also experiment with multiple batches on the belt assembly task to validate if the improvement is monotonic.

Finding: Large-batch DAgger is more suitable for training Compliant Residual Policy. The small-batch training becomes unstable and the demonstrator needs to provide large magnitudes of corrections as the number of iterations increases. During evaluation, the final policy always fails by inserting too high, while our single-batch policy achieves a 100% success rate with the same amount of data and training epochs.

Finding: More batches does not always bring better performance. Fig. 13 shows the change of success rate on the belt assembly task as the number of batches increases from one to three. The policy trained with three batches of data performs slightly worse than the one trained with two batches, indicating that more correction data does not always bring better performance. This could be caused by distribution shift as we observed different failure modes during each new batch of data collection, and the 50 additional episodes of data was not sufficient to cover the new distribution.

Sampling strategy during training. The start of a human intervention contains critical information of the timing and direction of correction. Accurate delta action predictions right after correction starts are important for reactive corrective behaviors. We investigate three strategies for sampling from online correction data during training: 1. Uniform sample, where the whole episode is sampled uniformly. 2. Denser sample around the start of a human intervention, and 3. denser sample only after the human intervention starts. For 2 and 3, we uniformly increase the sample frequency four times for a fixed period before and/or after intervention starts.

Finding: Sampling denser right after intervention starts leads to more reactive and accurate corrections. As shown in Fig. 12 (right), the best performance comes from densely sampling after the beginning of interventions. Sampling denser around the start of a human intervention also adds more samples right before the intervention starts, which is where humans observe signs of failures. These are mostly negative data, and using them for training decreases the policy success rate.

Intention Misinterpretation and tracking error. As explained in Sec. 3, the correction data can misinterpret the human correction behavior when tracking error is large. During development of our method, we observed this phenomenon on the gear insertion task, where the human correction motion is small in magnitude and mostly aligns with the base policy motion, both of which moves towards the pole. As a result, the residual policy learns to pull the gear away from the pole instead of moving it towards the pole, causing a persistent failure. With our fixes that reduces tracking error (equation 2 and 3), the correction data can correctly reflect the human intention and train a residual policy that improves success rate.

The problem does not affect the book flipping task and the belt assembly task even before our fixes are applied. For the book flipping task, the correction motion magnitude is typically much larger than the robot tracking error. The belt assembly task uses minute correction motion around the second pulley, however, the human correction motion is mostly vertical, in which direction the robot has little motion and tracking error.

5 Limitation and Future Work

Base/residual trade-off. The base policy should have a reasonable success rate for the residual policy to learn effectively. From our experiments, we recommend starting to collect correction data for the residual policy when the base policy has at least 10% $\sim$ 20% success rate. A practical question is whether to spend resources on improving the base policy or the residual policy, given a challenging new task. A future direction is to derive theoretical guidelines for such trade-off between the base and residual improvements.

Incorporate more data. Our work learns corrections efficiently using 50$\sim$100 episodes. The cost of this efficiency is the lack of extreme robustness under large variations, which does require more information about the problem dynamics. There are two interesting questions to ask for future work: 1) how can more data/information be absorbed by the residual policy to make it adapt to more diverse scenarios? Can we use a world model/dynamics model to guide the correction behavior under novel perturbations? An example work in this direction is Sun and Song (2025). 2) Is there better policy

architecture than a shallow MLP? Throughout this work, we use a MLP as the action head of our Compliant Residual Policy and directly regress the actions. We briefly explored popular alternatives such as a probabilistic MLP Duan et al. (2016); Haarnoja et al. (2018) or Flow Matching Lipman et al. (2022); Black et al. (2024) in Sec. A.4, but found no significant difference. Although a simple MLP already works well in our tasks, we believe it may experience difficulty for complex tasks that involve more distinctive action multi-modalities, for which more expressive policy formulations might be useful.

More flexible/intuitive data collection interface. Our data collection system is based on kinesthetic teaching. Although it provides richer data with higher quality than teleoperation as explained in the paper, it may require more labor during training data collection since the demonstrator needs to grasp the handle on the robot. Two interesting future work directions include 1) develop teleoperation systems with the same intuitive and smooth correction data collection, and 2) develop intuitive interface for manipulation with more DoFs, such as bi-manual or dexterous hands manipulation.

6 Conclusion

In this work, we evaluate practical design choices for DAgger in real-world robot learning, and provide a system, CR-DAgger, to effectively collect human correction data with a Compliant Intervention Interface and improve the base policy with a Compliant Residual Policy. We demonstrate the effectiveness of our designs by comparing them with a variety of alternatives on four contact-rich manipulation tasks.

Acknowledgements

We would like to thank Eric Cousineau, Huy Ha, and Benjamin Burchfiel for thoughtful discussions on the proposed method, thank Mandi Zhao, Maximillian Du, Calvin Luo, Mengda Xu, and all REALab members for their suggestions on the experiment setup and the manuscript.

Funding

This work was supported in part by the NSF Award #2143601, #2037101, and #2132519, Sloan Fellowship, and Toyota Research Institute. We would like to thank Google and TRI for the UR5 robot hardware. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies, either expressed or implied, of the sponsors.

References

Shir Amir, Yossi Gandelsman, Shai Bagon, and Tali Dekel. Deep vit features as dense visual descriptors. *arXiv preprint arXiv:2112.05814*, 2(3):4, 2021.

Lars Ankile, Anthony Simeonov, Idan Shenfeld, Marcel Torne, and Pulkit Agrawal. From imitation to refinement–residual rl for precise assembly. *arXiv preprint arXiv:2407.16677*, 2024.

Philip J Ball, Laura Smith, Ilya Kostrikov, and Sergey Levine. Efficient online reinforcement learning with offline data. In *International Conference on Machine Learning*, pages 1577–1594. PMLR, 2023.

Maria Bauza, Antonia Bronars, Yifan Hou, Ian Taylor, Nikhil Chavan-Dafle, and Alberto Rodriguez. Simple, a visuotactile method learned in simulation to precisely pick, localize, regrasp, and place objects. *Science Robotics*, 9(91):eadi8808, 2024.

Homanga Bharadhwaj, Roozbeh Mottaghi, Abhinav Gupta, and Shubham Tulsiani. Track2act: Predicting point tracks from internet videos enables generalizable robot manipulation. In *European Conference on Computer Vision*, pages 306–324. Springer, 2024.

Kevin Black, Noah Brown, Danny Driess, Adnan Esmail, Michael Equi, Chelsea Finn, Niccolo Fusai, Lachy Groom, Karol Hausman, Brian Ichter, et al. π_0: A vision-language-action flow model for general robot control. *arXiv preprint arXiv:2410.24164*, 2024.

Yuhui Chen, Shuai Tian, Shugao Liu, Yingting Zhou, Haoran Li, and Dongbin Zhao. Conrft: A reinforced fine-tuning method for vla models via consistency policy. *arXiv preprint arXiv:2502.05450*, 2025.

Cheng Chi, Zhenjia Xu, Siyuan Feng, Eric Cousineau, Yilun Du, Benjamin Burchfiel, Russ Tedrake, and Shuran Song. Diffusion policy: Visuomotor policy learning via action diffusion. *The International Journal of Robotics Research*, page 02783649241273668, 2023.

Cheng Chi, Zhenjia Xu, Chuer Pan, Eric Cousineau, Benjamin Burchfiel, Siyuan Feng, Russ Tedrake, and Shuran Song. Universal manipulation interface: In-the-wild robot teaching without in-the-wild robots. *arXiv preprint arXiv:2402.10329*, 2024.

Hojung Choi, Jun En Low, Tae Myung Huh, Gabriela A Uribe, Seongheon Hong, Kenneth AW Hoffman, Julia Di, Tony G Chen, Andrew A Stanley, and Mark R Cutkosky. Coinft: A coin-sized, capacitive 6-axis force torque sensor for robotic applications. *arXiv preprint arXiv:2503.19225*, 2025.

Yan Duan, Xi Chen, Rein Houthooft, John Schulman, and Pieter Abbeel. Benchmarking deep reinforcement learning for continuous control. In *International conference on machine learning*, pages 1329–1338. PMLR, 2016.

Ying Fan, Olivia Watkins, Yuqing Du, Hao Liu, Moonkyung Ryu, Craig Boutilier, Pieter Abbeel, Mohammad Ghavamzadeh, Kangwook Lee, and Kimin Lee. Dpok: Reinforcement learning for fine-tuning text-to-image diffusion models. *Advances in Neural Information Processing Systems*, 36:79858–79885, 2023.

Ian J Goodfellow, Mehdi Mirza, Da Xiao, Aaron Courville, and Yoshua Bengio. An empirical investigation of catastrophic forgetting in gradient-based neural networks. *arXiv preprint arXiv:1312.6211*, 2013.

Irmak Guzey, Yinlong Dai, Ben Evans, Soumith Chintala, and Lerrel Pinto. See to touch: Learning tactile dexterity through visual incentives. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 13825–13832. IEEE, 2024a.

Irmak Guzey, Yinlong Dai, Georgy Savva, Raunaq Bhirangi, and Lerrel Pinto. Bridging the human to robot dexterity gap through object-oriented rewards. *arXiv preprint arXiv:2410.23289*, 2024b.

Tuomas Haarnoja, Aurick Zhou, Pieter Abbeel, and Sergey Levine. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International conference on machine learning*, pages 1861–1870. Pmlr, 2018.

Siddhant Haldar, Vaibhav Mathur, Denis Yarats, and Lerrel Pinto. Watch and match: Supercharging imitation with regularized optimal transport. In *Conference on Robot Learning*, pages 32–43. PMLR, 2023a.

Siddhant Haldar, Jyothish Pari, Anant Rai, and Lerrel Pinto. Teach a robot to fish: Versatile imitation from one minute of demonstrations. *arXiv preprint arXiv:2303.01497*, 2023b.

Zhanpeng He, Yifeng Cao, and Matei Ciocarlie. Uncertainty comes for free: Human-in-the-loop policies with diffusion models. *arXiv preprint arXiv:2503.01876*, 2025.

Neville Hogan. Impedance control: An approach to manipulation. In *1984 American control conference*, pages 304–313. IEEE, 1984.

Ryan Hoque, Ashwin Balakrishna, Ellen Novoseller, Albert Wilcox, Daniel S Brown, and Ken Goldberg. Thriftydagger: Budget-aware novelty and risk gating for interactive imitation learning. *arXiv preprint arXiv:2109.08273*, 2021a.

Ryan Hoque, Ashwin Balakrishna, Carl Putterman, Michael Luo, Daniel S Brown, Daniel Seita, Brijen Thananjeyan, Ellen Novoseller, and Ken Goldberg. Lazydagger: Reducing context switching in interactive imitation learning. In *2021 IEEE 17th international conference on automation science and engineering (case)*, pages 502–509. IEEE, 2021b.

Yifan Hou and Matthew T Mason. Robust execution of contact-rich motion plans by hybrid force-velocity control. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 1933–1939. IEEE, 2019.

Yifan Hou, Zhenzhong Jia, and Matthew Mason. Manipulation with shared grasping. In *Robotics: Science and Systems*, 2020.

Yifan Hou, Zeyi Liu, Cheng Chi, Eric Cousineau, Naveen Kuppuswamy, Siyuan Feng, Benjamin Burchfiel, and Shuran Song. Adaptive compliance policy: Learning approximate compliance for diffusion guided control. In *2025 IEEE International Conference on Robotics and Automation (ICRA)*, pages 4829–4836, 2025. doi: 10.1109/ICRA55743.2025.11128452.

Tobias Johannink, Shikhar Bahl, Ashvin Nair, Jianlan Luo, Avinash Kumar, Matthias Loskyll, Juan Aparicio Ojea, Eugen Solowjow, and Sergey Levine. Residual reinforcement learning for robot control. In *2019 international conference on robotics and automation (ICRA)*, pages 6023–6029. IEEE, 2019.

Michael Kelly, Chelsea Sidrane, Katherine Driggs-Campbell, and Mykel J Kochenderfer. Hg-dagger: Interactive imitation learning with human experts. In *2019 International Conference on Robotics and Automation (ICRA)*, pages 8077–8083. IEEE, 2019.

Alexander Khazatsky, Karl Pertsch, Suraj Nair, Ashwin Balakrishna, Sudeep Dasari, Siddharth Karamcheti, Soroush Nasiriany, Mohan Kumar Srirama, Lawrence Yunliang Chen, Kirsty Ellis, et al. Droid: A large-scale in-the-wild robot manipulation dataset. *arXiv preprint arXiv:2403.12945*, 2024.

Moo Jin Kim, Karl Pertsch, Siddharth Karamcheti, Ted Xiao, Ashwin Balakrishna, Suraj Nair, Rafael Rafailov, Ethan Foster, Grace Lam, Pannag Sanketi, et al. Openvla: An open-source vision-language-action model. *arXiv preprint arXiv:2406.09246*, 2024.

Kenneth Kimble, Karl Van Wyk, Joe Falco, Elena Messina, Yu Sun, Mizuho Shibata, Wataru Uemura, and Yasuyoshi Yokokohji. Benchmarking protocols for evaluating small parts robotic assembly systems. *IEEE robotics and automation letters*, 5 (2):883–889, 2020.

James Kirkpatrick, Razvan Pascanu, Neil Rabinowitz, Joel Veness, Guillaume Desjardins, Andrei A Rusu, Kieran Milan, John Quan, Tiago Ramalho, Agnieszka Grabska-Barwinska, et al. Overcoming catastrophic forgetting in neural networks. *Proceedings of the national academy of sciences*, 114(13):3521–3526, 2017.

Yaron Lipman, Ricky TQ Chen, Heli Ben-Hamu, Maximilian Nickel, and Matt Le. Flow matching for generative modeling. *arXiv preprint arXiv:2210.02747*, 2022.

Huihan Liu, Soroush Nasiriany, Lance Zhang, Zhiyao Bao, and Yuke Zhu. Robot learning on the job: Human-in-the-loop autonomy and learning during deployment. *The International Journal of Robotics Research*, page 02783649241273901, 2022.

Yun Liu, Xiaomeng Xu, Weihang Chen, Haocheng Yuan, He Wang, Jing Xu, Rui Chen, and Li Yi. Enhancing generalizable 6d pose tracking of an in-hand object with tactile sensing. *IEEE Robotics and Automation Letters*, 9(2):1106–1113, 2023.

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine. Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning. *arXiv preprint arXiv:2410.21845*, 2024.

Kevin M Lynch and Frank C Park. *Modern robotics*. Cambridge University Press, 2017.

Ajay Mandlekar, Danfei Xu, Roberto Martín-Martín, Yuke Zhu, Li Fei-Fei, and Silvio Savarese. Human-in-the-loop imitation learning using remote teleoperation. *arXiv preprint arXiv:2012.06733*, 2020.

Max Sobol Mark, Tian Gao, Georgia Gabriela Sampaio, Mohan Kumar Srirama, Archit Sharma, Chelsea Finn, and Aviral Kumar. Policy agnostic rl: Offline rl and online rl fine-tuning of any class and backbone. *arXiv preprint arXiv:2412.06685*, 2024.

Matthew T Mason. Compliance and force control for computer controlled manipulators. *IEEE Transactions on Systems, Man, and Cybernetics*, 11(6):418–432, 2007.

Abby O'Neill, Abdul Rehman, Abhiram Maddukuri, Abhishek Gupta, Abhishek Padalkar, Abraham Lee, Acorn Pooley, Agrim Gupta, Ajay Mandlekar, Ajinkya Jain, et al. Open x-embodiment: Robotic learning datasets and rt-x models: Open x-embodiment collaboration 0. In *2024 IEEE International Conference on Robotics and Automation (ICRA)*, pages 6892–6903. IEEE, 2024.

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

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

Jonathan Spencer, Sanjiban Choudhury, Matthew Barnes, Matthew Schmittle, Mung Chiang, Peter Ramadge, and Siddhartha Srinivasa. Learning from interventions: Human-robot interaction as both explicit and implicit feedback. In *16th robotics: science and systems, RSS 2020*. MIT Press Journals, 2020.

Zhanyi Sun and Shuran Song. Latent policy barrier: Learning robust visuomotor policies by staying in-distribution. *arXiv preprint arXiv:2508.05941*, 2025.

Octo Model Team, Dibya Ghosh, Homer Walke, Karl Pertsch, Kevin Black, Oier Mees, Sudeep Dasari, Joey Hejna, Tobias Kreiman,

Charles Xu, et al. Octo: An open-source generalist robot policy. *arXiv preprint arXiv:2405.12213*, 2024.

Aaron Van Den Oord, Sander Dieleman, Heiga Zen, Karen Simonyan, Oriol Vinyals, Alex Graves, Nal Kalchbrenner, Andrew Senior, Koray Kavukcuoglu, et al. Wavenet: A generative model for raw audio. *arXiv preprint arXiv:1609.03499*, 12, 2016.

Luigi Villani and Joris De Schutter. Force control. In *Springer handbook of robotics*, pages 195–220. Springer, 2016.

Philipp Wu, Yide Shentu, Qiayuan Liao, Ding Jin, Menglong Guo, Koushil Sreenath, Xingyu Lin, and Pieter Abbeel. Robocopilot: Human-in-the-loop interactive imitation learning for robot manipulation. *arXiv preprint arXiv:2503.07771*, 2025.

Haoyu Xiong, Xiaomeng Xu, Jimmy Wu, Yifan Hou, Jeannette Bohg, and Shuran Song. Vision in action: Learning active perception from human demonstrations. *arXiv preprint arXiv:2506.15666*, 2025.

Mengda Xu, Manuela Veloso, and Shuran Song. Aspire: Adaptive skill priors for reinforcement learning. *Advances in Neural Information Processing Systems*, 35:38600–38613, 2022.

Mengda Xu, Zhenjia Xu, Cheng Chi, Manuela Veloso, and Shuran Song. Xskill: Cross embodiment skill discovery. In *Conference on robot learning*, pages 3536–3555. PMLR, 2023a.

Mengda Xu, Zhenjia Xu, Yinghao Xu, Cheng Chi, Gordon Wetzstein, Manuela Veloso, and Shuran Song. Flow as the cross-domain manipulation interface. *arXiv preprint arXiv:2407.15208*, 2024a.

Xiaomeng Xu, Yanchao Yang, Kaichun Mo, Boxiao Pan, Li Yi, and Leonidas Guibas. Jacobinerf: Nerf shaping with mutual information gradients. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16498–16507, 2023b.

Xiaomeng Xu, Huy Ha, and Shuran Song. Dynamics-guided diffusion model for robot manipulator design. *arXiv preprint arXiv:2402.15038*, 2024b.

Xiaomeng Xu, Dominik Bauer, and Shuran Song. Robopanoptes: The all-seeing robot with whole-body dexterity. *arXiv preprint arXiv:2501.05420*, 2025.

Xiaomeng Xu*, Yifan Hou*, Chendong Xin, Zeyi Liu, and Shuran Song. Compliant residual dagger: Improving real-world contact-rich manipulation with human corrections. In *The 39th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2025.

Patrick Yin, Tyler Westenbroek, Simran Bagaria, Kevin Huang, Ching-an Cheng, Andrey Kobolov, and Abhishek Gupta. Rapidly adapting policies to the real world via simulation-guided fine-tuning. *arXiv preprint arXiv:2502.02705*, 2025.

Wenzhen Yuan, Siyuan Dong, and Edward H Adelson. Gelsight: High-resolution robot tactile sensors for estimating geometry and force. *Sensors*, 17(12):2762, 2017.

Xiu Yuan, Tongzhou Mu, Stone Tao, Yunhao Fang, Mengke Zhang, and Hao Su. Policy decorator: Model-agnostic online refinement for large policy model. *arXiv preprint arXiv:2412.13630*, 2024.

A Technical Appendices and Supplementary Material

A.1 Controller Architecture

Our software system consists of three independent loops:

1. **The base policy loop** that runs the diffusion policy. The base policy loop updates at about 1Hz, each time predicts 32 frames of actions corresponding to 3.2s of future robot positions. In this work we have not optimized the implementation for computation speed.

2. **The residual policy loop** that runs at approximately 50Hz, each time predicts five frames of delta actions corresponding to 0.1s of delta positions. The delta actions are added to the corresponding base policy actions based one time, before being sent to the low-level compliance controller for execution. The loop rate is limited by our current implementation and can be improved if needed.

3. **The admittance controller loop** that runs at exactly 500Hz. This loop implements 6D Cartesian compliance on the robot. It takes reference positions and forces from the residual policy. When there is zero reference force and no external force, the admittance controller lets the robot track the reference position precisely. When external force exists, the robot will deviate from the reference position like a spring centered on the base policy output position.

Apart from the above controller/policy loops, each hardware (e.g. camera, force-torque sensor) has a standalone driver loop maintaining 1. communication with the hardware, and 2. buffers for action and feedback for this hardware.

We choose admittance control to implement compliance in this work, since the Universal Robot we use has high position-tracking accuracy and is not back-drivable. We open-sourced our implementation at `https://github.com/yifan-hou/force_control`. A more through review of compliance controller implementations can be found in Lynch and Park (2017) and Villani and De Schutter (2016).

A.2 Stage-Wise Success Rate

We report the success rate of the book flipping task into three key stages. Fig. 14 below is a more detailed version of Fig. 9, which reports all the task success rates by stages.

A.3 Correction Data Decomposition

As mentioned in the "Training strategy" part of § 3.2, we used two strategies to ensure the residual policy behaves stably around low correction data regions. The first strategy is to include the no correction portion of online data for training and label them with zero residual actions. The second strategy is to collect a few trajectories (15 out of the 50 total correction episodes) in which the demonstrator marks the whole trajectory as correction, even when the correction is small or zero. In practice, we find that the first strategy works better when the base policy is more stable and has a higher success rate, while the second strategy works better otherwise. In our experiments, we use the first strategy for the book

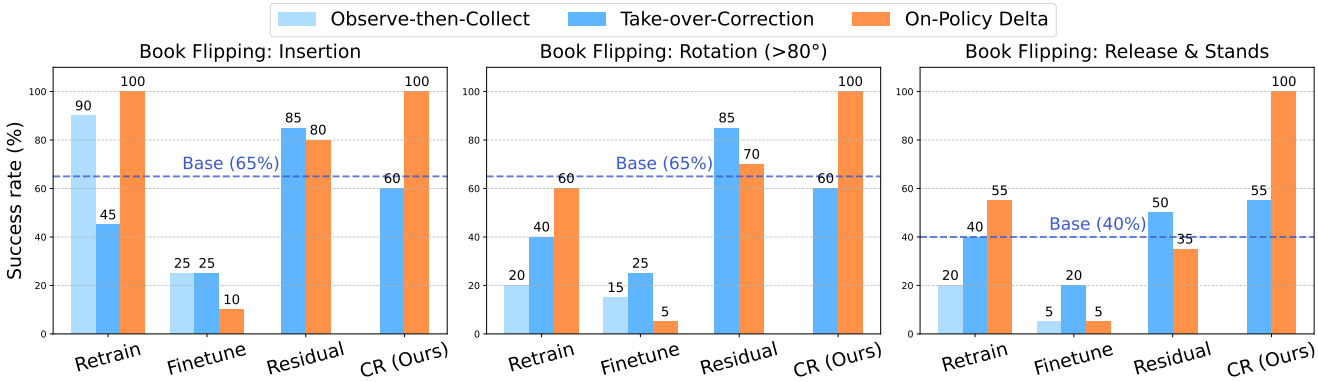

Figure 14. Stage-wise Success rates. Each row represents the results for one task, while each column shows the success rate up to the corresponding stage.

flipping task and use the second strategy for the other three tasks.

A.4 Comparison of Policy Structures

To further explore alternative residual policy formulations beyond the standard MLP action head used in our experiments, we tested two additional popular policy structures on the belt assembly task: a probabilistic MLP Duan et al. (2016); Haarnoja et al. (2018) and a Flow-Matching Transformer Lipman et al. (2022); Black et al. (2024). The probabilistic MLP extends the original deterministic MLP by outputting both the mean and log-variance of a Gaussian distribution over actions and sampling actions from this distribution. The Flow Matching Transformer uses a transformer-based conditional flow model that predicts action velocities along a continuous time trajectory, enabling ODE-based sampling over the residual action distribution instead of one-step prediction.

We trained residual policies with these two models using the same data under identical configurations as the original MLP structure on the belt assembly task. The probabilistic MLP achieves a success rate of 100%, and the Flow Matching Transformer achieves 90% (each ablation was evaluated over 10 rollouts). Compared to the success rate of 96.8% achieved by the original MLP, the improvements from more expressive policy structures are inconclusive. The failure modes in this task are relatively homogeneous. We hypothesis that the benefits of expressive policy structure could be more evident for tasks that exhibit more distinctive action multi-modality and leave the more detailed exploration to future work.

A.5 Experiments Compute Resources

We use a desktop with a NVIDIA GeForce RTX 4090 GPU for training and deployment.

A.6 Hardware Design

Our kinesthetic correction hardware setup features a tool interface that allows task-specific tool swapping. For the book flipping task, we designed a customized fork-shaped tool that can easily insert under the books and flip them. For the belt assembly task, we used a standard WSG-50 gripper and fin-ray fingers Chi et al. (2024). An interesting future direction is to leverage generative models for automatic manipulator design Xu et al. (2024b). Future work can also incorporate other types of force or tactile sensors, such as capacitive F/T sensors Choi et al. (2025) and vision-based tactile sensors Yuan et al. (2017); Liu et al. (2023); Bauza et al. (2024).