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

Xiaomeng Xu* Yifan Hou* Zeyi Liu Shuran Song
Stanford University

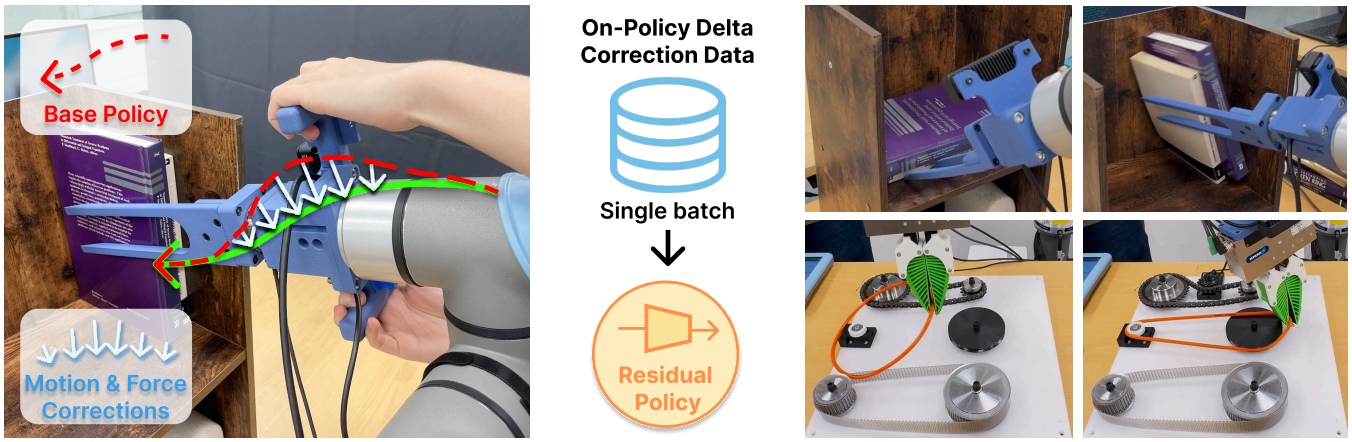

Fig. 1: **CR-DAgger.** To improve a robot manipulation policy, we propose a compliant intervention interface (a) for collecting human correction data, and use this data to update a compliant residual policy (b), and thoroughly study their effects by deploying the updated policy on two contact-rich manipulation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

***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 over 50% on two challenging tasks (book flipping and belt assembly) 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://cr-dagger.github.io/**

I. Introduction

To obtain a successful policy with learning from demonstration, 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) [1, 2]. However, doing DAgger effectively for real-world robotic problems still faces the following challenges:

How to collect informative human correction data? DAgger is most effective when the correction data is within the original policy's induced state distribution [1]. In practice, the common approach is either (1) collecting offline demonstrations that cover the policy's typical failure scenarios [3], or (2) human taking over robot control during policy deployment [4, 5]. However, in both cases, it is difficult for human demonstrators to gain the intuition for how to execute the corrective actions without deviating excessively from the original distribution. 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 [2], which can be computationally expensive; (2) finetuning the policy with only the additional data [6, 7, 8], which is sensitive to the quality of the new data [9], and (3) training a residual policy separately on top of the pretrained policy, which is typically done with Reinforcement Learning [10, 9] or Imitation Learning [11], 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 feel the magnitude of their instantaneous

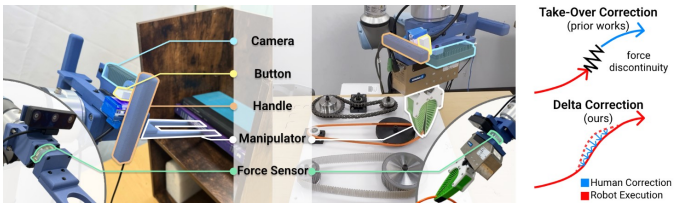

Fig. 2: **Compliant Intervention Interface** characterized by a kinesthetic correction hardware setup where humans hold on the handle and apply forces to correct robot execution, providing on-policy delta corrections.

correction, so they can provide gentle adjustments. Unlike take-over corrections that may cause force discontinuity, our design allows smooth transition between correction/no correction mode, while maintaining distributional consistency with the original policy.

- **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 two challenging tasks with long horizons and sequences of contacts: book flipping and belt assembly. We improve over the base policy success rate by over 50% using less than 50 intervention episodes, while also outperforming retrain-from-scratch and finetuning under the same data budgets.

II. CR-DAgger Method

Our goal is to improve a pretrained robot policy with a small amount of human correction data. Throughout the paper, we use the term *base policy* to refer to the pretrained policy without online improvements.

A. Compliant Intervention Interface

Correction data is most effective when it corrects failures in policy-induced state distributions [1]. 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 help correction data maintain distributional consistency with base policy outputs.

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 policy's original 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 intervention may still introduce significant distribution shifts as the motion deviates too much from the original policy 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 [7, 8]. 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, 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 that can easily lead to out-of-distribution states. 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

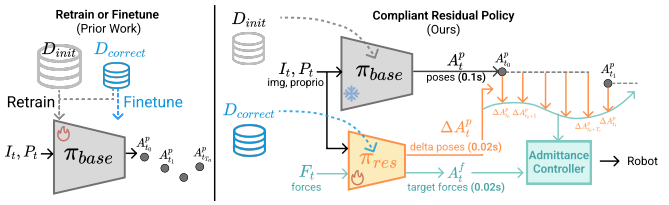

Fig. 3: **Policy Update Methods.** Left: Common policy update methods - retraining and finetuning. Right: Ours. The base policy runs at 1 Hz. It takes in images I_t and proprioceptions P_t and predicts 32 frames of end-effector poses $A_t^p = \{A_{t_0}^p, A_{t_1}^p, ..., A_{T_a}^p\}$ spaced 0.1 Seconds apart. The Compliant Residual Policy runs at 50 Hz. It takes in additional force inputs F_t and predicts 5 frames of delta poses $\Delta A_t^p = \{\Delta A_{t_0}^p, \Delta A_{t_0+1}^p, ..., \Delta A_{t_0+T_r}^p\}$ and target forces A_t^f spaced 0.02 Seconds apart. The combined poses of A_t^p and ΔA_t^p, and target forces A_t^f are taken by an admittance controller to command the robot.

different tasks. We run a compliance controller (specifically admittance control) in the background to respond to both contact forces and human correction forces, allowing the human to influence but not completely override the policy execution. 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.

B. 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 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. 3. 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 [3], 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 to extract an image embedding, a temporal convolution network [12] to encode the force vectors, followed by fully-connected layers to decode actions.

Advantages of this formulation include:

- *Sample-efficient learning.* The residual policy's network is light-weight and only requires a small amount of correction data to train ($\sim$50 demonstrations).
- *Incorporating new sensor modality.* 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.

Training strategy. In prior work, a residual policy is trained either in simulation with RL [10, 9] to give it sufficient coverage of the input distribution, or in the real world with pre-collected behavior cloning data [13]. 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.
- *Prioritize correction data over no-correction (zero residual action) data.* Similar to [14], we alter the sample frequency of intervention data 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 (§ III-E) demonstrate that our training strategies improve the quality of the residual policy and the overall success rate.

III. Evaluation

For each task, we train a diffusion policy [3] with 150 demonstrations as the base policy. We first deploy the base policy and observe its performance and failure modes. Next,

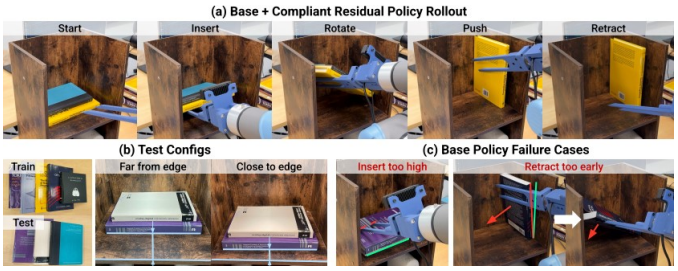

Fig. 4: **Book Flipping Task.** (a) Policy rollout of [Compliant Residual] policy trained with [On-Policy Delta] data, demonstrating accurate insertion motions and forceful pushing strategy. (b) Different test scenarios. (c) Typical failure cases of the base policy: inserting too high above the book and missing the gap; retracting the fingers before the books can steadily stand.

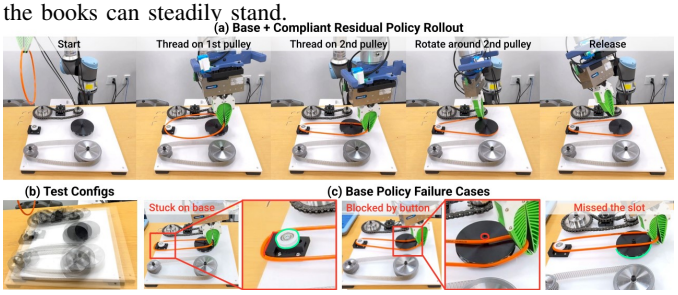

Fig. 5: **Belt Assembly Task.** (a) Policy rollout of [Compliant Residual] policy trained with [On-Policy Delta] data, demonstrating accurate force-position coordination and adaptation. (b) Different test scenarios. (c) Typical failure cases of the base policy: getting stuck on base next to the small pulley; blocked by the the button on the large pulley; and missing the slot by going too high above the pulley.

from the same base policy, we collect 50 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. Each data & policy combination is evaluated on each task for 20 trials. Details of tasks and comparisons are described below.

A. Contact-Rich Manipulation Tasks

Book Flipping: As shown in Fig. 4 (a), this task is to flip up books on a shelf with long, pointy fingers. The robot needs to first insert fingers below the book, then rotate and flip up the book, and finally push the book firmly against the shelf wall. A success is counted when the book can stand on its own after the fingers retract.

This task is challenging for its rich use of physical contacts and forceful strategies [15]. 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. 4 (b). We use the same test cases and initial configurations for all evaluations.

Belt Assembly: As shown in Fig. 5 (a), this task is to assemble a thin piece of belt onto two pulleys, which is part of the NIST board assembly challenge [16]. 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 task is counted as successful if the belt is stably assembled onto the two pulleys after its release.

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 resistant force and less position tolerance as the belt approaches the second pulley, requiring a policy with good force-position coordination and adaptation. We ran 20 rollouts across 4 different initial board positions for all methods (Fig. 5 (b)). We use the same test cases and initial robot configuration for all evaluations.

B. Base Policy and its Failure Modes

The book flipping base policy achieves a 40% success rate with the following common failure cases (Fig. 4 (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 20% success rate with the following common failure cases (Fig. 5 (c)): (1) Missed small pulley assembly. The policy often fails to properly control the height of the gripper, causing the belt to get stuck at the base of the board next to the small pulley. Sometimes the belt also gets stuck in the nearby pulleys due to misaligned planar positions. (2) Missed big pulley assembly. Common failure cases are the gripper either going above or below the big pulley, causing the belt to miss the slot.

C. Comparisons

We compare CR-DAgger with baselines across two dimensions: correction method and policy update method. We present the quantitative results in Fig. 6, and explain key findings in § III-D.

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.
- *Take-over-Correction* (HG-DAgger) [2] 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 robot policy does not affect the robot during correction. When correction ends

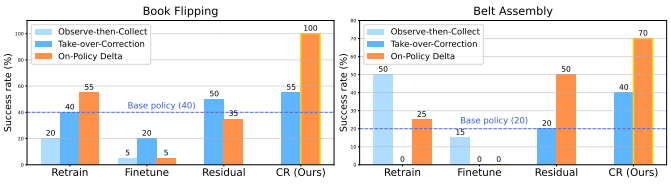

Fig. 6: **Results.** We compare CR-DAgger across two dimensions: correction method and policy update method. The result shows that our [Compliant Residual (CR)] policy trained with [On-Policy Delta] data is able to improve upon base policies on both tasks and outperforms other variations.

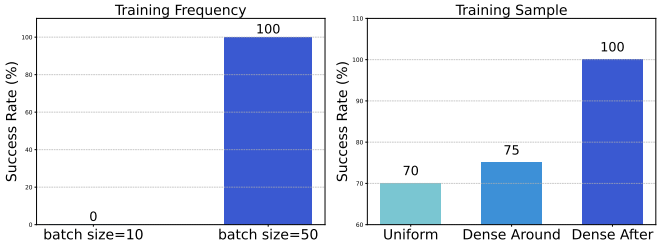

Fig. 7: **Effect of Training Frequency and Sample.** Single-batch update (batch size=50) leads to more stable training and dense sampling after correction starts achieves better performance.

(demonstrator released the button), we ask the demonstrator to keep holding the handle in place until the robot policy takes over again, so as to minimize the motion jitter caused by the switching.

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

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 § II-B, this approach is reliable but may require access to the orignal data and large amount of new data to work well.
- *Finetune Policy:* Finetune the base policy using only the correction data (freezing visual encoders). As explained in § II-B, this approach can be sensitive to data quality and distribution shifts.
- *Residual Policy:* an ablation of our method where force is removed from both input and outputs.
- *Complaint Residual Policy (Ours):* Residual policy update with additional force input and outputs, see details in § II-B.

D. Key Findings

Finding 1: Compliant Residual Policy is able to improve base policy by a large margin. As shown in Fig. 6, [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 touch the fingers down in the gap between the book and the shelf to increase the insertion accuracy; in the belt assembly task, the policy learns to adjust the height of the belt to thread it in the slot of the large pulley when initially misaligned. Results are best viewed in our supplementary video.

Finding 2: Residual policy allows additional useful modality to be added during correction. Force provides crucial information for contact-rich manipulation skills. [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 20% 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 second stage of the belt assembly task (threading the belt on the large pulley) requires delicate belt height adjustment based on its contact with the pulley, where visual information is ambiguous due to occlusions and the lack of depth. [Compliant Residual] policy exhibits reactive height adjustment behaviors - when the gripper moves short and touches the top of the pulley, the policy controls the gripper to move out along the pulley to find the slot for the belt.

Finding 3: Smooth On-Policy Delta data enables stable residual policy. [Compliant Residual] policy trained with [On-Policy Delta] data has 45% and 30% higher success rates than [Compliant Residual] & [Take-over-Correction] on the two tasks respectively. Both residual policies trained with [Take-over-Correction] data sometimes exhibit large noisy motions that trigger task failures. On the contrary, residual policies trained with [On-policy Delta] data have much smoother action trajectories and better reflect human's correction intentions, providing suitable magnitudes of corrections.

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 and 5% increase on the belt task, averaged across all data collection methods).

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 and 15% drop on the belt task, averaged across all data collection methods). This is likely due to the distribution mismatch between the base policy training distribution and correction data distribution, causing training instabilities.

E. Ablations

We study two important design decisions with ablation studies on the book flipping task.

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 distribution. However, DAgger with small batch sizes is known to suffer from *catastrophic forgetting* [17, 18] since it finetunes neural networks on data with non-stationary distribution. Common solutions include retraining the residual policy at the end of DAgger using all available correction data collected from all the intermediate residual policies [6].

Another way is to rely on the base policy training data as a normalizer [7]. In this work, we empirically found that a much simpler method works the best: single-batch training, *i.e.*, we train the residual only once. All correction data is obtained when only running the base policy. We compare our single-batch (batch size = 50) DAgger with a small batch size version, where we warm up the residual with 20 episodes of initial correction data, then update every ten more episodes for three times.

Finding: Single-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.

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 and staying in distribution. 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. 7 (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.

IV. Conclusion and Discussion

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 two contact-rich manipulation tasks.

Limitations and Future Work. 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 future direction is to derive theoretical guidelines for the trade-off between the base and residual improvements.

Throughout this work, we use a MLP as the action head of our Compliant Residual Policy and directly regress the actions. Although it works well in our tasks, it may experience difficulty for tasks that involve more distinctive action multi-modalities. More expressive policy formulations, such as Flow Matching [19, 20] might be useful for these tasks.

References

[1] 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.

[2] 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.

[3] 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.

[4] 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.

[5] 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.

[6] 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.

[7] 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.

[8] 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.

[9] 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.

[10] 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.

[11] 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.

[12] 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.

[13] 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.

[14] 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.

[15] Yifan Hou, Zhenzhong Jia, and Matthew T Mason. Manipulation with shared grasping. *arXiv preprint arXiv:2006.02996*, 2020.

[16] 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.

[17] 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.

[18] 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.

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

[20] 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.