Coherent Off-Policy Improvement of Large Behaviour Models with Learned Rewards

Anonymous Authors[1]

Abstract

Distilling expert demonstration data into large generative models using behavioural cloning is a scalable approach to learning capable policies for robotic control, particularly for dexterous manipulation. Reinforcement learning (RL) can be used as a means to fine-tune these policies further using additional experience. An open question is whether RL is more sample-efficient than collecting more human demonstrations. Prior work has fine-tuned large pre-trained policies in a scalable fashion by applying RL to a smaller residual policy that corrects the pre-trained model. However, for the typical sparse reward tasks, RL algorithms can struggle to optimize the behaviour in a sample-efficient manner. We look at inverse reinforcement learning, where a dense reward function is learned from the expert demonstrations, potentially reducing the challenge of RL fine-tuning. We specifically consider coherent imitation learning, an IRL method that facilitates improvement of the BC policy through using a specific reward formulation with theoretical guarantees. We show that our IRL method improves the performance of $\pi_0.5$ on all 6 sparse manipulation tasks and achieves a $\geq 90\%$ success rate on 5 out of 6 complex manipulation tasks, outperforming RL-based baselines using sparse rewards. By ensuring our initial pretrained finetuning policy is optimal for our initial reward and critic, our method circumvents the initial drop commonly seen in RL fine-tuning and enables faster improvement.

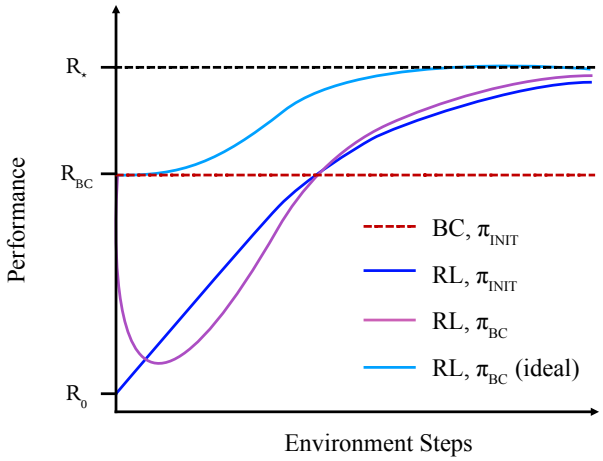


Figure 1. An illustration of RL fine-tuning of strong BC policies using residuals. Despite strong initial performance by a BC policy, due to function approximation in actor-critic methods, this performance is rapidly unlearned and relearned, which means in practice the BC initialization provides little benefit when running the RL method from scratch. We investigate a means for fine-tuning the BC policy without unlearning using coherent soft imitation learning (CSIL) for (inverse) RL fine-tuning.

1. Introduction

Distilling massive expert demonstration datasets into large generative models (behavioural cloning, BC) has seen large success in robotic control, yielding highly capable policies for complex, long-horizon manipulation (Driess et al., 2023; Barreiros et al., 2025). Despite these advances, covariate shift due to accumulating errors remains a fundamental limitation of any BC policy. Solutions such as collecting supplementary recovery demonstrations are fundamentally limited because they rely on teleoperators to anticipate out-of-distribution states. Since online agents will inevitably face novel states, online adaptation becomes indispensable.

Reinforcement learning (RL) is a natural candidate to improve BC policies through online experience. Particularly, residual RL offers an attractive set of characteristics for online adaptation: freezing the base policy and learning a correction allows for parameter-efficient fine-tuning of even large models. Recent efforts to improve online adaptation have introduced fine-grained corrections for action chunks

[1]Anonymous Institution, Anonymous City, Anonymous Region, Anonymous Country. Correspondence to: Anonymous Author <anon.email@domain.com>.

Preliminary work. Under review by the Decision-Making from Offline Datasets to Online Adaptation (DEMO) ICML Workshop 2026. Do not distribute.

(Ankile et al., 2025b), structured exploration (Yuan et al., 2025), and more efficient value learning (Ma et al., 2026). Despite these advances, two major challenges remain for RL for foundation models: First, designing dense, task-specific rewards for diverse, open-world tasks is difficult and prone to misspecification (Eschmann, 2021). Even in parameter-efficient paradigms like residual RL (Xiao et al., 2026), where the base model is frozen, the framework remains vulnerable to initial performance degradation (see Figure 1). Crucially, this unlearning directly degrades sample efficiency.

To achieve online adaptation without degrading the foundation model (see Figure 1), we build on coherent soft imitation learning (CSIL) (Watson et al., 2023), an entropy-regularized IRL algorithm. In contrast to other (deep) IRL algorithms, CSIL is designed to fine-tune a pre-trained behaviour policy using IRL due to its specially designed reward structure, as IRL methods with black-box reward models tend to unlearn the BC behaviour rather than improve it (Watson et al., 2023).

Our main contribution is a novel method for fine-tuning large behaviour models (LBMs) based on off-policy inverse RL. Rather than relying on the environment's rewards, which are typically sparse in manipulation tasks, we learn a dense reward function from demonstrations using CSIL, an inverse RL algorithm. Our second contribution is enhancing the base CSIL implementation (Watson et al., 2023) with recent sample-efficient off-policy RL advancements. This integration boosts image-based performance from 50% to 90% for `Square` and accelerates convergence speed in the state-based setting significantly.

2. Related Work

Achieving reliable performance with LBMs remains a critical bottleneck for their widespread deployment, as current pre-trained models typically fail to achieve sufficiently high success rates without further fine-tuning. While simple BC provides a baseline, a large body of recent literature focuses on improving LBMs through RL. Several works directly apply online RL algorithms, particularly proximal policy optimization (PPO), to fine-tune these models (Lu et al., 2025). However, the inherent sample inefficiency of these methods makes them difficult to scale and extend to real-world applications.

To overcome the cost of vast online exploration, offline RL has emerged as a powerful alternative for fine-tuning LBMs (Chen et al., 2026). These approaches typically train an offline value function and subsequently condition the LBM's actions upon it. While demonstrating impressive results, such methods are highly compute-intensive and generally out of scope for standard fine-tuning tasks. Seeking to mitigate both sample inefficiency and high computational demands, prior work has investigated off-policy residual RL (Ankile et al., 2025b;a). In this paradigm, RL is used to train a secondary policy that predicts corrections to the base actions of a pre-trained diffusion policy. Residual RL has recently been successfully adopted to LBMs (Xiao et al., 2026). Notably, the authors report a severe "initial performance drop" during early fine-tuning (Xiao et al., 2026), which is also observed in (Ankile et al., 2025a), driven by the agent's exploration into out-of-distribution states. The observed degradation is a well-documented phenomenon in the offline-to-online RL literature (Kostrikov et al., 2022; Watson et al., 2023; Yue et al., 2024). To prevent unlearning, these methods typically restrict action evaluation to the offline dataset or explicitly align their online objectives with the BC prior to avoid destructive, high-variance gradients.

Because the naive residual approach often suffers from poor exploration, recent works have introduced targeted improvements to this framework. For instance, constraining the residual predictions to a smaller, bounded action range has been shown to guide exploration more effectively (Yuan et al., 2025). Additionally, to improve and stabilize the value function's learning process, works have proposed filling the initial replay buffer exclusively with the base policy's actions (Ma et al., 2026).

Our approach contrasts itself in a variety of ways. Instead of training a residual, we ensemble the base policy's and the second policy's actions, as constraining the model to predict small residual corrections led to a collapse of uncertainty estimation in preliminary experiments. Initializing our policy using BC enables us to fill the initial replay buffer with on-policy actions, instead of relying on off-policy actions for this step (Ma et al., 2026). Most importantly, we learn a dense reward function in a principled manner in contrast to works that only use a sparse reward (Ankile et al., 2025b;a; Yuan et al., 2025; Ma et al., 2026) or approximate a reward signal using hand-crafted features (Lu et al., 2025). Further, we address sample-efficiency by incorporating advances from state-of-the-art off-policy RL such as categorical critics with batch normalization and weight normalization (Palenicek et al., 2025).

3. Coherent Inverse Reinforcement Learning

In this section, we will introduce the foundations of our method that are applicable independent of refining LBMs and, in particular, we will explain CSIL (Watson et al., 2023) and implementation details that improve its sample efficiency in the deep IRL setting.

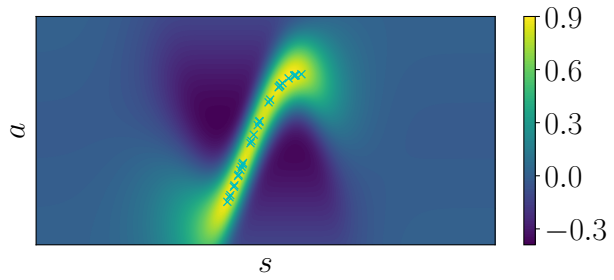


Figure 2. Given demonstration data (X), the coherent reward provides positive rewards for correct actions for observed states, negative rewards for incorrect actions for seen states, and zero rewards for any action under unseen states. This encourages the agent to act like the expert, and also rewards the agent for returning to the demonstration distribution if the agent deviates far from the expert trajectories due to compounding errors. Taken from Watson et al. (Watson et al., 2023).

3.1. Coherent Soft Imitation Learning

Imitation learning is an umbrella term that covers behavioural cloning and inverse reinforcement learning. Inverse reinforcement learning jointly performs RL and reward inference from demonstrations to learn expert behaviour. For sample efficiency, one would ideally combine these methods to accelerate learning, but in the deep RL setting, Watson et al. observed that most IRL algorithms cannot fine-tune BC policies and instead rapidly unlearn them (Watson et al., 2023). This pathology arises from function approximation. Since the BC policy is not optimal for the randomly initialized reward function and critic, a random behaviour is initially learned instead. To alleviate this issue, Watson et al. propose *coherency* in IRL, which requires that the BC policy be optimal for the learned reward function. In the entropy-regularized RL setting, this can be achieved with the following 'coherent' reward function,

$$\tilde{r}_\alpha(\mathbf{s}, \mathbf{a}) = \alpha \left(\log \pi_{\mathrm{BC}}(\mathbf{a}|\mathbf{s}) - \log \pi_{\mathcal{U}}(\mathbf{a}|\mathbf{s})\right), \qquad (1)$$

where π_{BC} is the pre-trained BC policy and $\pi_{\mathcal{U}}$ is uniform prior policy used for maximum entropy regularization. The ratio is further scaled by a parameter $\alpha > 0$, which is set to be scale invariant w.r.t. the action dimension. A notable property of this coherent reward is that it is a 'shaped' variant of the true unknown reward function, and therefore has the same optimal policy (Ng et al., 1999; Watson et al., 2023). More intuitively, the reward function specified in (1) should reward state-action pairs for which the state and action are in-distribution, should penalise state-action pairs where the state is in-distribution and the action is not and should return a reward of 0 if the state is out-of-distribution, independent of the action. To achieve the desired reward function behaviour, it is required that π_α models the data $\mathcal{D}$ when in-distribution and models the uniform prior otherwise. In practice, π_{BC} is usually a function approximator. As function approximators represent correlations between states, updating their approximation for a subset of states usually changes the approximation for a large set of other states as well. As the reward function's desired behaviour requires π_α to match the prior for out-of-distribution states, the authors introduce a MLP architecture that is 'stationary' that approximates this quality. Meronen et al. showed that a wide final layer with a periodic activation function induces the stationary property into the MLP (Meronen et al., 2021). Watson et al. specify a stationary policy $a = \tanh(z(\mathbf{s}))$, where $z(\mathbf{s}) = \mathbf{W}\phi(\mathbf{s})$, for which the weights are factorized row-wise $\mathbf{W} = [\mathbf{w}_1, ..., \mathbf{w}_{d_\alpha}]^\mathsf{T}$, $\mathbf{w}_i = \mathcal{N}(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$ to define a Gaussian Process with independent actions (Watson et al., 2023). By using change-of-variables as in SAC (Haarnoja et al., 2018), the authors express the policy per-action as

$$\phi(\mathbf{s}) = f_{per}(\tilde{\mathbf{W}}\phi_{mlp}(\mathbf{s})),$$

$$q(\mathbf{a}_i|\mathbf{s}) = \mathcal{N}\left(z_i; \boldsymbol{\mu}_i^\mathsf{T}\phi(\mathbf{s})\phi(\mathbf{s})^\mathsf{T}\boldsymbol{\Sigma}_i\phi(\mathbf{s})\right) \cdot \left|\det\left(\frac{\mathrm{d}a_i}{\mathrm{d}z_i}\right)\right|^{-1},$$

where f_{per} is a periodic activation function (Meronen et al., 2021), the weights $\tilde{\mathbf{W}}$ are drawn from a distribution, e.g., a Gaussian that characterises the stochastic process, and ϕ_{mlp} is an arbitrary MLP (Watson et al., 2023). They refer to this heteroscedastic stationary model as HETSTAT. The critic used for SAC is then initialised via

$$\tilde{Q}_1(\mathbf{s}, \mathbf{a}) = \tilde{r}_{\boldsymbol{\theta}_1}(\mathbf{s}, \mathbf{a}) + \gamma\big(\tilde{Q}_1(\mathbf{s}', \mathbf{a}') - \alpha(\log q_{\boldsymbol{\theta}_1}(\mathbf{a}' \mid \mathbf{s}') - \log \pi(\mathbf{a}' \mid \mathbf{s}'))\big),$$

where $\mathbf{s}, \mathbf{a}, \mathbf{s}', \mathbf{a}' \sim \mathcal{D}$ and $(\mathbf{s}', \mathbf{a}')$ is the successor state-action pair to $(\mathbf{s}, \mathbf{a})$. The reward's weights are then separated from the policy's weights such that both can be updated independently, and the policy and critic are updated using soft actor-critic (SAC) (Haarnoja et al., 2018).

For MDPs with success-based absorbing states, like many manipulation tasks, IRL methods can struggle if their rewards are positive. Agents are incentivized to 'stay alive' and accrue reward, and therefore never complete the task and enter the absorbing state. To prevent this behaviour in CSIL, the coherent reward can be bounded and its value subsequently shifted by its maximum reward, to ensure the coherent reward is always negative. To achieve this bounding of the maximum reward, the authors add a small bias term $\sigma^2_{\min}$ to the predictive variance that enables them to define an upper bound for the likelihood. As the $\tanh$ transformation is also bounded in practice to ensure numerical stability, the authors are then able to derive the following upper bound for the reward $r(\mathbf{s}, \mathbf{a})$, given action dimension d_a, a uniform prior across the action range $[-1, 1]^{d_a}$ and $\alpha = d_a^{-1}$

$$r(\mathbf{s}, \mathbf{a}) \leq -0.5 \cdot \log \pi\sigma^2_{min}/2 + \tilde{c},$$

where $\tilde{c}$ depends on the $\tanh$ clipping term (Watson et al., 2023).

3.2. Accelerating Coherent Learning

While Watson et al. showed CSIL demonstrated strong empirical results, the implementation struggled to solve more complex tasks in the pixel-based setting (Watson et al., 2023). To improve on CSIL's base performance, we integrated recent advances from the current state-of-the-art sample-efficient off-policy RL methods, which incorporate architectural regularization to facilitate more graceful scaling. In particular, following the results from Palenicek et al. (Palenicek et al., 2025), we adapt the CSIL critic and policy to use batch normalization (BN) (Ioffe & Szegedy, 2015) and weight normalization (WN) (Lyle et al., 2024) instead of layer normalization (Ba et al., 2016). Combining batch normalization and weight normalization improves the landscape of the optimization and helps maintain a steady 'effective' learning rate (Lyle et al., 2024; Palenicek et al., 2025). We observed that Watson et al.'s CSIL implementation, which is based on SAC, suffered from exploding parameter values during training, which impedes learning as the 'effective' learning rate drops. Incorporating BN and WN into CSIL allowed us to mitigate these numerical issues and improve optimization and scaling. Incorporating this regularization also enables us to increase the update-to-data (UTD) ratio, where the critic is updated several times per environment step to accelerate learning. In accordance with the results from (Fujimoto et al., 2018), we used policy delay and only updated our policy once per environment step. Additionally, while Watson et al. use a Monte Carlo KL estimator for the BC regularization term in the actor objective, we identify that this KL can be computed exactly between the latent Gaussian actions due to the shared bijection between policies (Polyanskiy, 2020). To improve CSIL's performance in image-based settings, we adapted CSIL's original impala-style encoder (Espeholt et al., 2018) by increasing its size, training an encoder per camera instead of one for all cameras and utilizing spatial softmax pooling (Levine et al., 2016) at the end of the network to extract structural information efficiently. We refer to this improved implementation of CSIL as CSIL++ in the subsequent text. With CSIL++ as our policy improvement operator, we now consider how to improve LBMs effectively.

4. Coherent fine-tuning of Pre-trained Policies

This section briefly outlines how CSIL is adapted to improve pre-trained action-chunking policies, in particular LBMs.

One important design decision is the RL action. This action could be the environment action, residual correction term

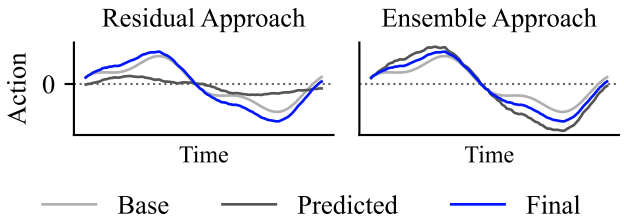


Figure 3. Visualization of the different action modalities. The light gray line is the action generated by the base policy, the dark gray is the one predicted by the trainable policy, and the blue line is the final action used in the environment. The final action for the residual approach is calculated via Eq. 2 while the total action using the ensemble approach is calculated via Eq. 3.

to the LBM (Eq. 2), or an ensembled policy, averaging the base action and the predicted action (Eq. 3),

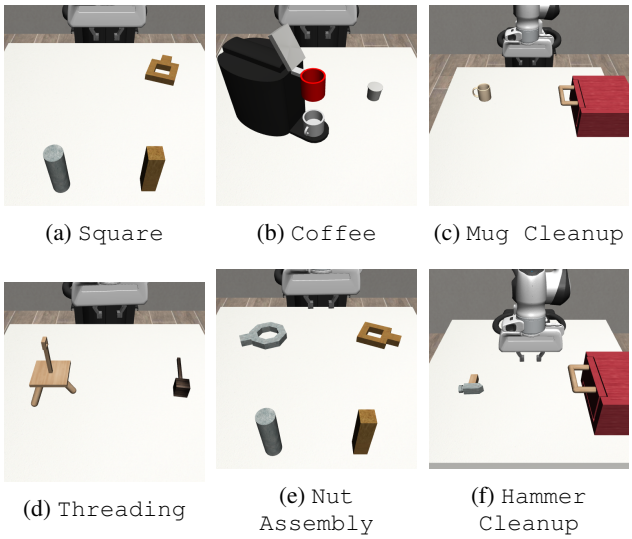
$$\boldsymbol{a}_{\text{residual}} = \boldsymbol{a}_{\text{base}} + \boldsymbol{a}_{\pi}, \tag{2}$$

$$\boldsymbol{a}_{\text{ensemble}} = \tfrac{1}{2}(\boldsymbol{a}_{\text{base}} + \boldsymbol{a}_{\pi}). \tag{3}$$

See Figure 3 for a visualization of both. Directly predicting environment actions, i.e., adapting the base policy directly, would require optimizing billions of parameters with RL, which is typically a prohibitive amount of compute even when using LoRA (Hu et al., 2022). It would also require RL over action chunks rather than single actions. To learn an additional, smaller corrective policy, we use the LBM to predict an action chunk every N steps (in our case $N = 10$) and a smaller model to predict a full action every step. Using a residual action, i.e., predicting a correction (Silver et al., 2018) on top of the base LBM policy, performed worse. We believe that this is because the residual prediction problem is challenging to learn as it is similar to learning random noise, rather than a more 'meaningful' mapping from observation to expert action, especially as we are jointly learning internal feature representations from vision. To incorporate 'meaning' back into the action prediction problem, we propose an ensemble approach where the smaller RL policy is averaged with the LBM action. Moreover, rather than using the LBM's internal representation for RL, we found it better and faster to let the RL policy learn its own observation encoder, which is smaller than the LBM's and therefore easier to fine-tune.

5. Experimental Setup

We test our approach on tasks from Robomimic (Mandlekar et al., 2021) and Mimicgen (Mandlekar et al., 2023), which vary in terms of task horizons, precision requirements and number of subtasks to be completed. Both Robomimic and Mimicgen rely on the Robosuite environment framework (Zhu et al., 2020), which is built on MuJoCo (Todorov et al., 2012). All tasks are run at a control frequency of 20 Hz. The main section of our exper-



(a) `Square` (b) `Coffee` (c) `Mug Cleanup`

(d) `Threading` (e) `Nut Assembly` (f) `Hammer Cleanup`

Figure 4. A visualization of the 6 manipulation tasks. In `Square`, the robot is tasked to place the wooden ring on the Square. `Nut Assembly` requires doing the same for the wooden ring and the silver ring. `Coffee` requires the capsule to be placed inside the machine and the lid to be closed. `Mug Cleanup` requires opening the drawer, putting the mug inside, and closing it, while `Hammer Cleanup` requires doing the same with the hammer. `Threading` requires picking up the needle and threading it through the tiny hole on top of the stick.

iments relies on an image-based setting, where the policy has access to a static camera and wrist-mounted camera, as well as the proprioceptive state of the end-effector. For our CSIL++ ablation, we also used a state-based setting, where the robot has access to ground-truth information about the environment, such as the current pose of the relevant object, as well as the end-effector's proprioceptive state. Our non-CSIL-based approaches use a constant negative reward of -1, and all our experiments utilize the successful termination for bootstrapping. We train all methods across 3 independent seeds for 500k steps and evaluate the policy every 50k steps for 50 episodes. To compare the maximum performance of the approaches, we store the parameters with the highest evaluation success rate and evaluate them 5 times, with 50 episodes each, at the end of training for each run. We use $\pi_{0.5}$ (Intelligence et al., 2025) as our base model with 10 actions per action chunk.

Tasks. Our task selection covers three categories: simple short-horizon manipulation tasks, dexterous, short-horizon tasks and mid-range horizon tasks. For simple short-horizon tasks, we use Coffee from Mimicgen (see Figure 4 for environment visualizations). For dexterous tasks, we use `Square` from Robomimic and `Nut Assembly` and `Threading` from Mimicgen. For mid-range horizon tasks, we use `Hammer Cleanup` and `Mug Cleanup`. All tasks use a Franka robot with a parallel-jaw gripper and have an action space dimension of 7 — 6 for the delta end-effector pose, and 1 for the gripper. For `Square` we use the released expert demonstration dataset with 200 episodes and for all Mimicgen tasks, we use the Mimicgen pipeline to construct a dataset of 200 episodes from a source dataset of 10 episodes.

Pre-training. We fine-tune $\pi_{0.5}$ for 30k steps with a batch size of 128, combining all single-task datasets into one dataset, for 13 hours on 8 A100 GPUs.

Baselines and Ablations. First, we examine how CSIL++ on top of an action-chunked base policy compares to applying off-policy RL with sparse rewards while including expert demonstrations into the training loop. For this, we chose the state-of-the-art algorithm XQC (Palenicek et al., 2026), an off-policy approach utilizing a categorical critic with batch normalization between the hidden layers and furthermore normalizes the weights of the hidden layers to the unit sphere after every gradient step. To incorporate expert demonstrations into the training loop, we modify XQC to concatenate a batch samples from the expert dataset to the batch sampled from the online replay buffer. Additionally we strengthen the XQC baseline, by incorporating BC pre-training and the same `HETSTAT` architecture used in our approach. The main differences of our approach are thus its dense reward and replacing the entropy regularization with a KL-Divergence based regularization. To ensure a fair comparison to current literature, we evaluate both using the policy trained via XQC + offline data (XQC+OD) as a residual as well as averaging the actions from the frozen base policy and the action predicted by the policy (XQC+OD Ensemble). Since all evaluated methods' policies are initialized via BC, our replay buffer is well initialized during training, as in (Ma et al., 2026). Additionally, we quantify the combined impact of our modifications on CSIL by comparing CSIL++ to vanilla CSIL on `Square` in both image-based and state-based settings.

Implementation Details and Hyperparameters. We summarize the specific architectural choices and hyperparameters used across our experiments in Table 2.

6. Experimental Results

We have introduced both the components of our method that are independent of fine-tuning pre-trained policies and those that are specific to fine-tuning pre-trained models and will now evaluate the performance of our approach. In particular, we will consider two questions:

1. How does CSIL++ compare to CSIL?
2. Can CSIL++ improve the performance of an LBMs?

We focus on sparse manipulation tasks in Robomimic and MimicGen and use their existing demonstration datastets.

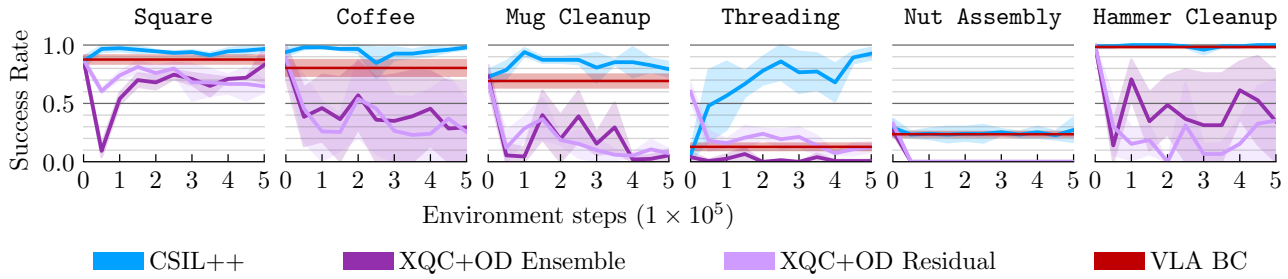


Figure 5. Performance of CSIL++ and other baselines on 6 challenging simulation environments across three seeds. While CSIL++ combines the coherent reward with the XQC actor-critic algorithm, XQC+OD uses the demonstration data and sparse environment reward. Success rate is computed over 50 evaluations.

While π_0 can solve the simpler Robomimic tasks to 100% success rates, the more complex tasks are not fully solved with the existing demonstration dataset of 200-1000 demonstrations.

6.1. Coherent fine-tuning of Large Behaviour Models

To evaluate our method, we conduct experiments on six environments from robomimic (Mandlekar et al., 2021) and mimicgen (Mandlekar et al., 2023). Examining Fig 5, we first observe that CSIL++ converges to a very strong policy for almost all tasks except `Nut Assembly`.

When analyzing the training progression, we find that both baselines exhibit a detrimental loss of performance at the beginning of each run. While both the ensemble baseline and the residual baseline recover some performance for simpler tasks like `Square` and `Coffee`, they do not stably reclaim the performance achieved by $\pi_{0.5}$ on its own. In particular, this can be observed for tasks with a longer horizon, such as `Nut Assembly`, `Hammer Cleanup`, and `Mug Cleanup`, where the models are not able to recover their initial performance. We attribute this strong decline to the lack of regularization against moving away from the BC policy. On the other hand, CSIL++ is able to circumvent this initial drop through its main benefits: The *coherence* of policy, reward, and critic detailed in 3 prevents an immediate and severe degradation of performance commonly seen in RL and inverse RL. The dense reward formulation allows the critic to have a more accurate estimate of the value, and the closed-form KL-Divergence regularization against the BC policy allows CSIL to bias the search space towards searching for solutions close to the approximated expert policy learned via BC.

These benefits also allow CSIL++ to keep the near-perfect performance on `Hammer Cleanup` and significantly improve on the strong $\pi_{0.5}$ base performance for `Square`, `Coffee`, and `Mug Cleanup`. For `Square` and `Coffee`, our approach achieves a near 100% success rate within 50k steps, while for `Mug Cleanup` it requires around 100k steps. Furthermore, for `Threading`, our approach improves the base policy's performance from roughly 13% to 89%, demonstrating its potential for improving the performance and robustness of base policy even in the low success rate range. For `Square` and `Mug Cleanup`, a slight drop after the initial performance peak is observable. We believe this is caused by the categorical critic being unable to discern between rewards for expert and online states due to the discretization caused by the binning once performance becomes near-perfect leading to the policy deviating too far from the BC policy. For `Nut Assembly`, we observe that our approach only preserves $\pi_{0.5}$'s performance but does not improve it. Likely this is caused by the sequential nature of the task, requiring to first solve the task of `Square` and then solve a similar task again, making it much harder to gather valuable samples for the second part of the task where it's most crucial to correct compounding errors from the base model.

Despite this drop, the success rate of the best-performing parameters for each seed, reported in Table 1, demonstrates that our approach is able to outperform the base policy's performance as well as the other baselines for all tasks ex-

Method	`Square`	`Coffee`	`Mug Cleanup`	`Threading`	`Nut Assembly`	`Hammer Cleanup`
CSIL++ Ensemble	**1.00 [0.99, 1.00]**	**0.98 [0.97, 0.98]**	**0.90 [0.81, 0.90]**	**0.90 [0.88, 0.91]**	0.32 [0.27, 0.35]	**1.00 [1.00, 1.00]**
XQC+OD Ensemble	0.85 [0.84, 0.85]	**0.98 [0.83, 0.99]**	0.74 [0.60, 0.82]	0.07 [0.04, 0.11]	0.27 [0.26, 0.28]	0.99 [0.98, 0.99]
XQC+OD Residual	0.94 [0.94, 0.94]	0.88 [0.87, 0.89]	0.80 [0.80, 0.80]	0.62 [0.62, 0.62]	**0.40 [0.40, 0.40]**	**1.00 [1.00, 1.00]**
VLA Base	0.84 [0.82, 0.86]	0.78 [0.76, 0.86]	0.68 [0.66, 0.72]	0.14 [0.12, 0.14]	0.22 [0.22, 0.26]	0.98 [0.98, 0.98]

Table 1. Success rates of best performing parameters in the evaluation during RL on 6 challenging manipulation tasks in simulation. **Bold** indicates highest median. Success rates are computed from 5 evaluations with 50 episodes each, and aggregate results across 3 seeds per environment and method are concatenated, reported as median, 25th and 75th percentiles

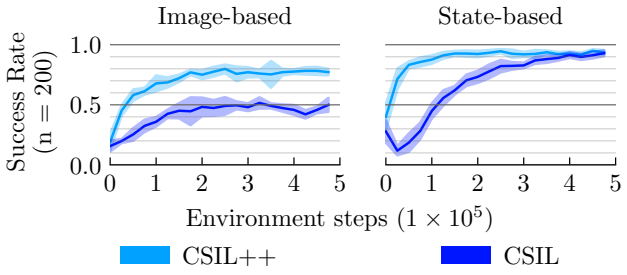


Figure 6. Performance of CSIL and our improved version CSIL++ on the `Square` task. The 'Image-based' run takes images and the robot's proprioceptive state as observations. The 'State-based' operates on full state information comprising proprioception and ground-truth information of the object. In both settings, CSIL++ is more sample-efficient and improves performance significantly in the 'Image-based' setting.

cept `Nut Assembly`. Notably, the strongest performing parameters for each baseline run are the initial BC parameters, showing the detrimental effect of the missing *coherence* in policy and critic and the lack of regularization against moving too far away from the BC policy.

6.2. Scaling Coherent Imitation Learning

To evaluate our modifications, we compare CSIL against our enhanced version, CSIL++, over 500k steps ($\sim 2,000$ rollouts) on the robomimic Square task in the state-based setting, where the actor has access to the robot's proprioceptive state as well as ground truth information about the object, as well as in the image-based setting, where the actor has access to a static camera, a wrist mounted camera and the robot's propriocepive state.

In Figure 6, we are able to see that CSIL++ reaches 85% within roughly 50k steps ($\sim$ 200 rollouts) in the state-based setting while vanilla CSIL takes around 250k steps, showing a 5x improvement in sample efficiency. The image-based performance gains are even more pronounced: CSIL++ reaches 70% success in 50k steps and converges to 90% by 150k steps, whereas vanilla CSIL reaches only 50% after 500k steps. In summary, our modifications lead to strictly superior performance in terms of sample efficiency and in the image-based setting, even in absolute performance.

7. Discussion

We demonstrate that utilizing inverse reinforcement learning problem to produce a coherent, dense reward resolves the severe sample inefficiency and instability typically associated with refining LBMs using RL. By replacing sparse environment signals with a dense reward derived from expert demonstrations we enable significant policy improvement. A key insight from our experiments is that the standard residual objective creates an overly complex optimization landscape for BC-initialized policies. Our ensembling approach sidesteps this by predicting full single-step actions and averaging them with the LBM's chunked actions while allowing the policy to learn a residual action by predicting actions deviating from the optimal action to compensate for the LBM's errors.

The primary limitation of our approach is its reliance on the smaller policy's ability to adequately capture the state distribution during the initial BC phase. For highly complex or sequential tasks with large initial state distributions, the smaller model struggles to learn a sufficient prior. This limits the ensemble's ability to correct compounding errors in later stages of the trajectory, as the model cannot gather enough valuable online samples to learn from. Additionally, our method currently only implicitly integrates the vast prior knowledge of the LBM at the action level, without leveraging the base model's rich semantic understanding as input for its predictions.

Looking ahead, a promising direction to overcome these limitations is to share knowledge more explicitly between the models. Routing the LBM's intermediate embeddings directly into the smaller policy could dramatically improve the smaller model's expressiveness and contextual awareness. Furthermore, incorporating inductive biases, such as SE(3) equivariance, into the smaller policy's architecture could help it generalize across much broader initial state distributions, allowing this coherent fine-tuning framework to scale to more complex, multi-stage manipulation tasks.

References

Ankile, L., Jiang, Z., Duan, R., Shi, G., Abbeel, P., and Nagabandi, A. Residual off-policy RL for finetuning behavior cloning policies. *arXiv preprint arXiv:2509.19301*, 2025a.

Ankile, L., Simeonov, A., Shenfeld, I., Torne, M., and Agrawal, P. From imitation to refinement-residual RL for precise assembly. In *IEEE International Conference on Robotics and Automation (ICRA)*, 2025b.

Ba, J. L., Kiros, J. R., and Hinton, G. E. Layer normalization, 2016.

Barreiros, J., Beaulieu, A., Bhat, A., Cory, R., Cousineau, E., Dai, H., Fang, C.-H., Hashimoto, K., Irshad, M. Z., Itkina, M., et al. A careful examination of large behavior models for multitask dexterous manipulation. *arXiv preprint arXiv:2507.05331*, 2025.

Chen, K., Liu, Z., Zhang, T., Guo, Z., Xu, S., Lin, H., Zang, H., Li, X., Zhang, Q., Yu, Z., Fan, G., Huang, T., Wang, Y., and Yu, C. π_{RL}: Online RL fine-tuning for flow-based vision-language-action models, 2026.

Driess, D., Xia, F., Sajjadi, M. S., Lynch, C., Chowdhery, A., Ichter, B., Wahid, A., Tompson, J., Vuong, Q., Yu, T., et al. PaLM-E: An embodied multimodal language model. In *International Conference on Machine Learning (ICML)*, 2023.

Eschmann, J. Reward function design in reinforcement learning. *Reinforcement learning algorithms: Analysis and Applications*, 2021.

Espeholt, L., Soyer, H., Munos, R., Simonyan, K., Mnih, V., Ward, T., Doron, Y., Firoiu, V., Harley, T., Dunning, I., Legg, S., and Kavukcuoglu, K. IMPALA: Scalable distributed deep-RL with importance weighted actor-learner architectures. In *International Conference on Machine Learning (ICML)*, 2018.

Fujimoto, S., van Hoof, H., and Meger, D. Addressing function approximation error in actor-critic methods, 2018.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *International Conference on Machine Learning (ICML)*, 2018.

Hu, E. J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., and Chen, W. LoRA: Low-rank adaptation of large language models. In *International Conference on Learning Representations (ICLR)*, 2022.

Intelligence, P., Black, K., Brown, N., Darpinian, J., Dhabalia, K., Driess, D., Esmail, A., Equi, M., Finn, C., Fusai, N., Galliker, M. Y., Ghosh, D., Groom, L., Hausman, K., Ichter, B., Jakubczak, S., Jones, T., Ke, L., LeBlanc, D., Levine, S., Li-Bell, A., Mothukuri, M., Nair, S., Pertsch, K., Ren, A. Z., Shi, L. X., Smith, L., Springenberg, J. T., Stachowicz, K., Tanner, J., Vuong, Q., Walke, H., Walling, A., Wang, H., Yu, L., and Zhilinsky, U. $\pi_{0.5}$: a vision-language-action model with open-world generalization, 2025.

Ioffe, S. and Szegedy, C. Batch normalization: Accelerating deep network training by reducing internal covariate shift, 2015.

Kostrikov, I., Nair, A., and Levine, S. Offline reinforcement learning with implicit q-learning. In *International Conference of Learning Representation (ICLR)*, 2022.

Levine, S., Finn, C., Darrell, T., and Abbeel, P. End-to-end training of deep visuomotor policies. *Journal of Machine Learning Research (JMLR)*, 2016.

Lu, G., Guo, W., Zhang, C., Zhou, Y., Jiang, H., Gao, Z., Tang, Y., and Wang, Z. VLA-RL: Towards masterful and general robotic manipulation with scalable reinforcement learning, 2025.

Lyle, C., Zheng, Z., Khetarpal, K., Martens, J., van Hasselt, H., Pascanu, R., and Dabney, W. Normalization and effective learning rates in reinforcement learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2024.

Ma, G., Li, L., Wang, H., Liu, Z., Bacon, P.-L., and Tao, D. What makes value learning efficient in residual reinforcement learning? *arXiv preprint arXiv:2602.10539*, 2026.

Mandlekar, A., Xu, D., Wong, J., Nasiriany, S., Wang, C., Kulkarni, R., Fei-Fei, L., Savarese, S., Zhu, Y., and Martín-Martín, R. What matters in learning from offline human demonstrations for robot manipulation. In *Conference on Robot Learning (CoRL)*, 2021.

Mandlekar, A., Nasiriany, S., Wen, B., Akinola, I., Narang, Y., Fan, L., Zhu, Y., and Fox, D. MimicGen: A data generation system for scalable robot learning using human demonstrations. In *Conference on Robot Learning (CoRL)*, 2023.

Meronen, L., Trapp, M., and Solin, A. Periodic activation functions induce stationarity. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2021.

Ng, A. Y., Harada, D., and Russell, S. J. Policy invariance under reward transformations: Theory and application to reward shaping. In *International Conference on Machine Learning (ICML)*, 1999.

Palenicek, D., Vogt, F., Watson, J., and Peters, J. Scaling off-policy reinforcement learning with batch and weight normalization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2025.

Palenicek, D., Vogt, F., Watson, J., Posner, I., and Peters, J. XQC: Well-conditioned optimization accelerates deep reinforcement learning. In *International Conference of Learning Representation (ICLR)*, 2026.

Polyanskiy, Y. Information theoretic methods in statistics and computer science: Lecture 1 — f-divergences, 2020.

Silver, T., Allen, K., Tenenbaum, J., and Kaelbling, L. Residual policy learning. *arXiv preprint arXiv:1812.06298*, 2018.

Todorov, E., Erez, T., and Tassa, Y. MuJoCo: A physics engine for model-based control. In *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)*, 2012.

Watson, J., Huang, S. H., and Heess, N. Coherent soft imitation learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023.

Xiao, W., Lin, H., Peng, A., Xue, H., He, T., Xie, Y., Hu, F., Wu, J., Luo, Z., Fan, L., et al. Self-improving vision-language-action models with data generation via residual rl. In *International Conference of Learning Representation (ICLR)*, 2026.

Yuan, X., Mu, T., Tao, S., Fang, Y., Zhang, M., and Su, H. Policy decorator: Model-agnostic online refinement for large policy model. In *International Conference on Learning Representations (ICLR)*, 2025.

Yue, S., Hua, X., Ren, J., Lin, S., Zhang, J., and Zhang, Y. OLLIE: imitation learning from offline pretraining to online finetuning. In *International Conference on Machine Learning (ICML)*, 2024.

Zhu, Y., Wong, J., Mandlekar, A., Martín-Martín, R., Joshi, A., Nasiriany, S., Zhu, Y., and Lin, K. robosuite: A modular simulation framework and benchmark for robot learning. In *arXiv preprint arXiv:2009.12293*, 2020.

A. Hyperparameter

Table 2. Core Hyperparameters for CSIL++

Hyperparameter	**Value**
Architecture & Optimization	
Batch Size	128
Base Network Sizes	Same as CSIL (Watson et al., 2023)
Learning Rates	Same as CSIL (Watson et al., 2023)
ResNet Channels	$[16, 32, 64, 128]$
Visual Pooling	Spatial Softmax
Critic Normalization	Batch Normalization
Policy Normalization	Layer Normalization
Categorical Critic Bins	101
Categorical Critic Range	$[-8, 0]$
Distribution of Bins	By Quantile
Training Dynamics	
Warmup Steps	10,000
Total Environment Steps	510,000
Update-To-Data (UTD) Ratio	4
Entropy Coefficient	0.3
Replay Buffer Size	250,000
Pretraining	
Demonstrations per Env.	200
Policy Pretraining Steps (Image)	25,000
Policy Pretraining Steps (State)	50,000
Critic Pretraining Steps	20,000
Environment Horizon	Standard (Task-dependent)