ARTIFICIAL INTELLIGENCE

Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning

Jianlan Luo*, Charles Xu*, Jeffrey Wu, Sergey Levine

Robotic manipulation remains one of the most difficult challenges in robotics, with approaches ranging from classical model-based control to modern imitation learning. Although these methods have enabled substantial progress, they often require extensive manual design, struggle with performance, and demand large-scale data collection. These limitations hinder their real-world deployment at scale, where reliability, speed, and robustness are essential. Reinforcement learning (RL) offers a powerful alternative by enabling robots to autonomously acquire complex manipulation skills through interaction. However, realizing the full potential of RL in the real world remains challenging because of issues of sample efficiency and safety. We present a human-in-the-loop, vision-based RL system that achieved strong performance on a wide range of dexterous manipulation tasks, including precise assembly, dynamic manipulation, and dual-arm coordination. These tasks reflect realistic industrial tolerances, with small but critical variations in initial object placements that demand sophisticated reactive control. Our method integrates demonstrations, human corrections, sample-efficient RL algorithms, and system-level design to directly learn RL policies in the real world. Within 1 to 2.5 hours of real-world training, our approach outperformed other baselines by improving task success by 2×, achieving near-perfect success rates, and executing 1.8× faster on average. Through extensive experiments and analysis, our results suggest that RL can learn a wide range of complex vision-based manipulation policies directly in the real world within practical training times. We hope that this work will inspire a new generation of learned robotic manipulation techniques, benefiting both industrial applications and research advancements.

INTRODUCTION

Manipulation is one of the foundational problems in robotics, and achieving human-level performance on dynamic, dexterous manipulation tasks is a longstanding pursuit in the field (*1*). Traditional approaches have shown promise but face substantial limitations: Hand-designed controllers attain high performance for narrowly defined tasks but typically lack adaptability to new scenarios, whereas imitation learning (IL) methods often sacrifice speed and robustness when faced with environmental variations. Reinforcement learning (RL) is a promising approach for autonomous acquisition of complex and dexterous robotic skills. By learning through trial and error, an effective RL method should, in principle, be able to acquire highly proficient skills tailored to the particular physical characteristics of the deployment task. This could result in performance that not only exceeds that of hand-designed controllers but also surpasses that of human teleoperation. However, realizing this promise in real-world settings has been challenging because of issues with sample complexity, assumptions (e.g., accurate reward functions), and optimization stability. RL methods have been effective for training in simulation (*2*–*6*) and for training on existing large real-world datasets with the aim of broad generalization (*7*, *8*). They have also been used with hand-designed features or representations for narrowly tailored tasks (*9*, *10*). However, it remains challenging to develop general-purpose, vision-based methods that can efficiently learn physically complex skills across diverse real-world tasks and exceed the proficiency of IL and hand-designed controllers at the same time. We believe that making fundamental progress on this front can unlock new opportunities, which will then boost the development of truly performant robotic manipulation policies.

Here, we develop an RL system for vision-based manipulation that can acquire a wide range of precise and dexterous robotic skills. Our system, named human-in-the-loop sample-efficient robotic reinforcement learning (HIL-SERL), addresses the previously mentioned challenges by integrating a number of components that enable effective vision-based RL policies in the real world. To address the optimization stability issue, we used a pretrained visual backbone for policy learning. To handle the sample complexity issue, we used a sample-efficient off-policy RL algorithm based on RL with prior data (RLPD) (*11*) that incorporates human demonstrations and corrections. In addition, a well-designed low-level controller was included to ensure safety during policy training. During training, the system queried a human operator for potential corrections, which were then used to update the policy in an off-policy manner. We found that this human-in-the-loop correction procedure is crucial in enabling the policy to learn from its mistakes and improve performance, particularly for challenging tasks considered in this paper, which are hard to learn from scratch.

As shown in Fig. 1, the tasks our system solves include dynamically flipping an object in a pan; whipping out a Jenga block from a tower; handing over objects between two arms; and assembling complex devices such as a computer motherboard, an IKEA shelf, a car dashboard, or a timing belt using either one or two robotic arms. These tasks present challenges in terms of complex and intricate dynamics, high-dimensional state and action spaces, long horizons, or combinations thereof. Some of these skills were previously considered infeasible to train with RL directly in real-world settings, such as many of the dual-arm manipulation tasks, or nearly insurmountable with current robotic methods, like timing belt assembly or Jenga whipping. In addition, they require different types of control strategies, such as reactive closed-loop control for precise manipulation

Department of Electrical Engineering and Computer Sciences, University of California, Berkeley, CA 94720, USA.
*Corresponding author. Email: jianlanluo@berkeley.edu (J.L.); xuc@berkeley.edu (C.X.)

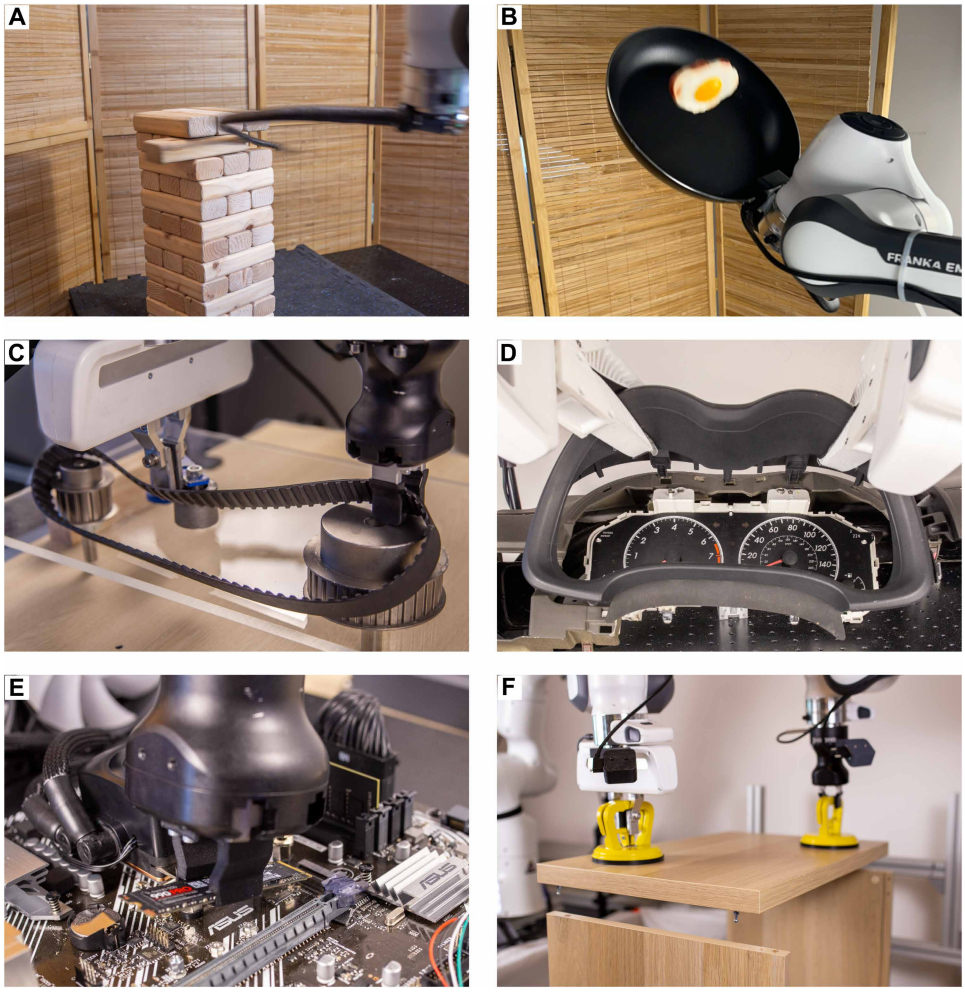


Fig. 1. Overview of experimental tasks. A subset of tasks considered in this paper, including (**A**) whipping out a Jenga block from its tower, (**B**) flipping an object in a pan, and assembling complex devices such as (**C**) a timing belt, (**D**) a dashboard, (**E**) a motherboard, and (**F**) an IKEA shelf.

tasks or delicate open-loop behaviors that are otherwise very difficult to prescribe, e.g., Jenga whipping. However, perhaps the most unexpected finding is that our system can train RL policies to achieve near-perfect success rates and superhuman cycle times on almost all of the tasks with only 1 hour to 2.5 hours of training time in the real world, subject to a few centimeters or degrees of variation in initial placement. Our trained RL policies greatly outperformed IL methods that were trained on the same amount of human data, e.g., the same number of episodes of demonstrations or corrections, 101% improvement in terms of success rate on average, and 1.8× faster cycle times. The result is important because it demonstrates that RL can learn a wide range of complex vision-based manipulation policies directly in the real world within practical training times, which was previously considered infeasible with earlier methods. Moreover, RL does so with a superhuman level of performance that greatly exceeds that of IL and hand-designed controllers.

To assess the effectiveness of our system, we compared it against several state-of-the-art RL methods and conducted ablation studies to understand the contribution of each component. The results demonstrate that our system not only outperforms the relevant baselines but also highlights that the impressive empirical results are due to the careful integration of these components. Furthermore, we provide a comprehensive analysis of the empirical results, offering insights into the effectiveness of RL-based manipulation. This analysis explores the training dynamics of the learned RL policies and further examines the flexibility of RL policies to serve as a general-purpose vision-based policy for acquiring distinct types of control strategies.

In summary, our contributions demonstrate that with the appropriate system-level design choices, RL can effectively solve a wide range of dexterous and complex vision-based manipulation tasks in the real world. Our system enables dual-arm coordination from image inputs and handles tasks such as whipping out a Jenga block and assembling a timing belt, demonstrating the flexibility of this approach across diverse manipulation scenarios. We further provide a comprehensive analysis of the empirical success of RL-based manipulation, offering insights into the effectiveness of RL-based manipulation. This analysis shapes our understanding of why RL succeeds in these complex tasks and suggests potential directions for further extending RL-based manipulation to even more challenging scenarios. For the accompanying code and videos, please refer to our Supplementary Materials. The proposed system uses RL to solve dexterous manipulation tasks; thus, we survey related works on real-world robotic RL methods and systems as well as other approaches that address similar dexterous manipulation tasks.

Dexterous robotic manipulation

For some of the tasks considered in this paper, prior works have explored alternative approaches. In insertion tasks, prior works have used model-based approaches (*12*, *13*) and end-effector tooling mechanisms with passive compliance (*14*, *15*). These methods often rely on state-based models without perception or require task-specific development, limiting robustness and adaptability. Sim-to-real transfer has been explored for some of the insertion tasks (*16*–*20*), offering easier data collection and the potential for broader generalization. However, these approaches often rely on privileged information—such as state or pose estimators—or operate directly on point cloud inputs to bridge the sim-to-real gap. In contrast, our work focuses on direct real-world RL using only visual inputs from multiple cameras. By training entirely on physical hardware, we avoided common sim-to-real challenges such as inaccuracies in contact dynamics, lighting, and sensor noise. Despite relying solely on real-world data and raw visual inputs, our system achieved high performance within a short, practical training time, often outperforming reported sim-to-real results on tasks of comparable

complexity. Another approach involves using visual servoing in a multistage pipeline to align the robotic arm with the target, followed by search primitives for insertions (*21*–*23*). Prior methods also face challenges with feature reliability and alignment precision. In contrast, our method uses a much tighter perception-action loop. It learns task-relevant visual features and visuomotor policies in a closed-loop manner, crucial for many of the reactive high-precision tasks. The learned policy can be viewed as an instance of output feedback control from the controls perspective (*24*).

There are a number of works tackling the dynamic manipulation tasks (*25*) considered in this paper. The authors of (*26*) used a motion capture system and dynamic motion primitives (*27*) to learn how to flip an object in the pan. However, our system directly uses pixel inputs, which alleviates the need for precise motion capture systems and achieve much higher success rates. The authors of (*28*) proposed a learning method to push out a Jenga block from its tower in a quasidynamic manner. Our approach, however, uses a whip to dynamically remove the Jenga block, presenting a more challenging task that requires a much more sophisticated control policy. Furthermore, although there are studies on flexible object manipulation, such as cable routing (*29*, *30*), tracing, or untangling (*31*–*33*), the timing belt assembly task in our paper demands reactive yet precise coordination between two arms to dynamically adjust both the tensioner and the timing belt. This task is fundamentally different and more challenging than that in previous works on cable manipulation.

Algorithms and systems for real-world RL

Real-world robotic RL requires algorithms that are sample efficient in handling high-dimensional inputs such as onboard perception and supporting easy specification of rewards and resets. Several algorithms have demonstrated the ability to learn efficiently directly in the real world (*34*–*48*). These include variants of off-policy RL (*43*, *49*, *50*), model-based RL (*51*–*55*), and on-policy RL (*56*). Despite progress, these often require long training times. Our system achieved superhuman performance on complex tasks with shorter training times. Other works have researched inference of rewards from raw visual observation through success classifiers (*57*, *58*), foundation model–based rewards (*59*–*61*), and rewards from videos (*62*, *63*). To enable autonomous training, there have been a number of algorithmic advances in reset-free learning (*64*–*68*) that require minimal human interventions during training. Although we do not introduce new algorithms in these areas, our framework effectively integrates existing methods. As detailed in Materials and Methods, using binary classifier–based rewards with demonstrations is effective for the complex tasks in this paper. One of the most relevant works to our research is SERL (*48*), which also presents a system for training RL policies for manipulation tasks. Our approach differs from SERL in that we incorporate both human demonstrations and corrections to train RL policies, whereas SERL relies solely on human demonstrations. Although this distinction might appear minor, our results indicate that integrating corrections is crucial for enabling the policy to learn from its mistakes and improve performance, especially for tasks that are challenging for the agent to learn from scratch. In addition, SERL focuses on simpler tasks with relatively short horizons and does not address dual-arm coordination or dynamic manipulation. Our unique contribution is demonstrating that our approach can effectively learn general-purpose vision-based manipulation policies across a wide range of tasks with varying physical characteristics, setting our system fundamentally apart from prior work on SERL. With the results presented in this paper, we hope that this work will serve as a stepping stone for future learning-based robotic manipulation research and, in the long term, enable robust deployable robotic manipulation skills capable of adapting to diverse environments and tasks, bringing us closer to the goal of general-purpose robotic manipulation.

RESULTS

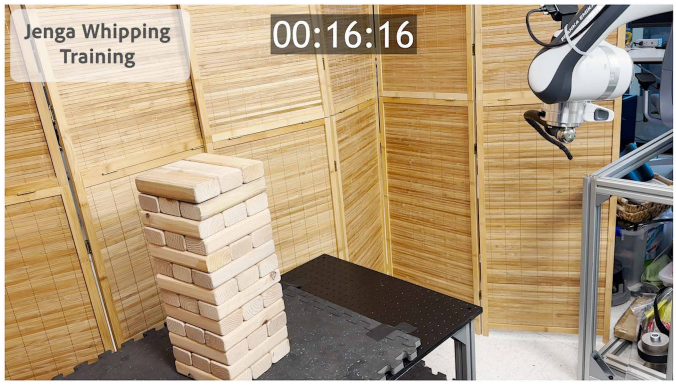
In this section, we discuss our experiments. We first present the experimental setup and the results. We then discuss these results and their implications. Movie 1 summarizes the method and findings of this work.

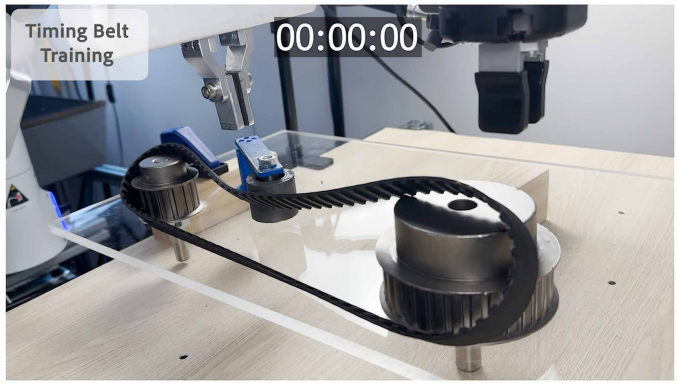
Overview of experiments

We conducted experiments across seven diverse tasks covering a range of distinct characteristics, as illustrated in Fig. 2. These tasks encompass a range of manipulation challenges, including dynamic object manipulation (e.g., flipping an object in a pan), precise and delicate manipulation [e.g., inserting a solid-state drive (SSD) into its matching slot], combined dynamic and precise manipulation (e.g., inserting a component while the target is moving), flexible object manipulation (e.g., assembling a timing belt), and multistage tasks with multiple subtasks (e.g., assembling an IKEA shelf). We solved these tasks by using either a single robot arm or a dual-arm setup, together with various combinations of observations and actions. The observation space could include images from wrist-mounted and side cameras, end-effector poses, twists, forces/torques, and the current gripper status of both arms. For dynamic tasks where the acceleration of the end effector was important, the action space directly commanded feedforward wrenches in the end-effector frame, which can be roughly thought of as desired accelerations. For other tasks that performed visual servoing behavior, the action space was each arm's six-dimensional (6D) Cartesian twist target for the downstream impedance controller. For tasks that required learned grasping, we also included discrete actions for controlling the grippers. For all tasks, unless otherwise noted, we trained a binary classifier as a reward detector, which took images from wrist and/or side cameras as inputs and predicted whether the current state was a success or failure for completing the current task. To train such classifiers, we collected both positive and negative demonstrations from human operators. We also collected additional potential false-positive or false-negative examples if necessary. We include details of the training process for each task in the Supplementary Materials. For tasks involving grasping, we also included a small negative penalty for gripper actions to discourage the policy from operating its grippers unnecessarily. Each task also used either a scripted robot motion or a manual human reset to randomize the initial state of the task. Details for the setup of each task and policy training can be found in the Supplementary Materials. In the remainder of this section, we first describe each task in detail and present relevant results as well as comparisons with other state-of-the-art methods.

Description of tasks

In this subsection, we present descriptions of the tasks in our experiments. We picked our tasks to cover a broad range of manipulation challenges, including contact-rich dynamics, dual-arm coordination, flexible object handling, and dynamic manipulation. Here, we organized the tasks such that similar challenges are presented together.





Movie 1. Summary of the method and representative results in this paper.

We first present two tasks that require precise manipulation in a contact-rich setting, followed by three tasks that require dual-arm coordination to solve hard tasks, including flexible object manipulation. We then proceed to two tasks that require dynamic manipulation. An illustration of each task can be found in Fig. 2.

Motherboard assembly

The motherboard assembly task includes four subtasks: inserting a random access memory (RAM) card into its matching slot, assembling a peripheral component interconnect express SSD into the motherboard, picking up a free-floating USB cable and inserting it into the slot, and securing the USB cable into a tight-fitting clip.

RAM insertion. In this task, the robot is supposed to insert a RAM card into the matching slot. The process involves two main steps: The robot first needs to align the RAM card with the narrow openings on both sides of the slot and then proceed with delicate downward motion with appropriate force to insert the RAM card into the slot. The RAM card is assumed to be pregrasped by the robot, although we also periodically placed it back to a fixture and regrasped it with 1 cm of randomization in the longitudinal direction to introduce grasping variations. The end effector's starting position was randomized in a 4 cm–by–4 cm region with 6° of rotation randomization in the z axis. The task was considered successful if the RAM card was fully inserted into the slot without triggering the locking mechanism, allowing for easy reset. If desired, an additional downward force could be applied after executing the trained policy to lock the RAM card in place. This task is challenging because applying slightly excessive force can cause the RAM card to tilt within the gripper, leading to failure, whereas insufficient force may result in the RAM card not being properly inserted into the slot.

SSD assembly. In this task, the robot was supposed to insert one side of the SSD into its matching slot and then place the other side onto the fixture residing in the motherboard. Like the RAM insertion task, the SSD was assumed to be pregrasped by the robot with 1 cm of randomization in the longitudinal direction. The end effector's starting position was randomized in a 2 cm–by–2 cm region with 1° of rotation randomization in the z axis. The task was considered successful if both sides of the SSD were properly mated to their counterparts. This task required a gentle but precise insertion strategy initially to avoid damaging the contact pins, followed by another precise motion to align the other side with the supporting fixture.

USB connector grasp-insertion. In this task, a USB cable was freely placed on a table, and the robot was supposed to grasp the USB connector part, insert it into the corresponding slot, and release the gripper. The cable was randomly dropped by the arm within a 2 cm–by–2 cm randomization region before the arm was moved to another random position within this region with 10° of rotation randomization in the z axis. This task was considered successful if the USB connector was fully inserted into the slot and the gripper was released. The difficulty lies in the variability in the initial placement of the USB cable, as does the uncertainty in the grasping pose; the policy must learn to account for such uncertainty during insertion. For example, if an unsuitable grasp was executed, then the policy might need to release the object and regrasp it to achieve a better grasp pose.

USB cable clipping. This task assumed that a USB cable is already plugged into the motherboard, and the robot was tasked to pick up the remaining part of the cable and insert it into a tight-fitting organization clip. The cable was randomly dropped by the arm from a 4 cm–by–4 cm region before the arm was moved to another random position within this region with 10° in rotation in the z axis. The task was considered successful if the USB cable was fully inserted into the clip. The difficulty lies in the variability of the deformable USB cable and the tight insertion phase.

The whole assembly. We also performed the whole assembly task by chaining the above four subtasks together, using scripted motions to transition between subtasks. Movie S1 shows the entire assembly process and demonstrates that the computer successfully booted up, validating the effectiveness of our method in not only achieving task success but also performing the RL training process gracefully without damaging these delicate components.

IKEA assembly

The IKEA assembly task involved assembling an IKEA shelf with four boards and was decomposed into three subtasks: The robot first needed to assemble the two side panels into a panel fixed on the table and then mount the top panels into side panels after they are assembled. The task was considered successful if all the pieces were properly assembled into the shelf. For all of the subtasks, we assumed that the panels were pregrasped by the robot; however, we did periodically place them back to fixtures and regrasp them to introduce grasping variations.

Side panel assembly. In this task, the top part of the side panel was assumed to be pregrasped by the robot, although the grasping location can vary because of the heavy weight of the panel during the interactive assembly process, and we did periodically regrasp it from the fixtures. The arms grasping the panel start with 8 cm–by–8 cm randomization in x- and y-axis translation and 1° in z-axis rotation.

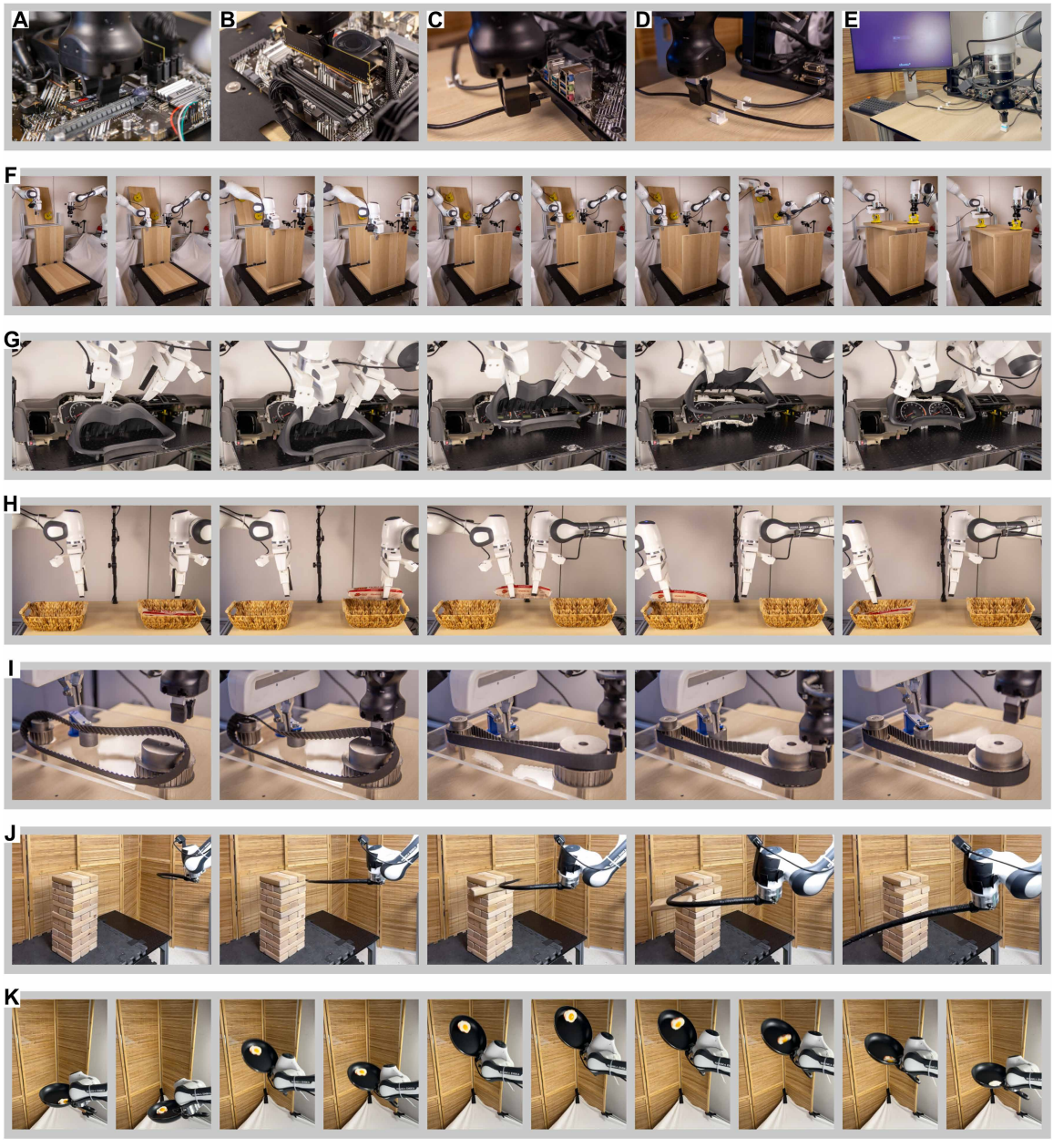


Fig. 2. Illustrations of the tasks in our experiments. (**A** to **E**) A sequence of motherboard assembly tasks: SSD installation, RAM insertion, USB cable grasping and insertion into a slot and a clip, and booting up the computer to ensure motherboard functionality. (**F**) A manipulation sequence to assemble IKEA furniture: The robot first assembles two side panels and then installs the top panel onto the mounted side panels. (**G**) A manipulation sequence to assemble a car dashboard. Two robot arms first grasp the workpiece and then align multiple pins to the slots. (**H**) Two arms performing a coordinated handover task. (**I**) Two arms performing a timing belt installation task. (**J**) A manipulation sequence of the Jenga whipping task, where the robot needs to extract one Jenga piece from the tower without crashing it. (**K**) The robot flips the object in the pan to the opposite side.

The task is considered successful if the bottom part is properly assembled onto the two matching pins.

Top panel assembly. After two side panels were assembled, this task demanded that the robot mount the top panels onto both of the side panels. The panel starts with 3 cm–by–3 cm randomization in x- and y-axis translation and 1° in z-axis rotation. The task was considered successful if all four pins of the top panel were properly inserted into the corresponding holes on the side panel. This task was difficult because the top parts of the side panels can move around during the assembly process, and the policy must adapt to these variations to succeed.

The whole assembly. We also performed the whole assembly task by chaining the three trained policies, using scripted motions to transition between subtasks, which is illustrated in Fig. 2 and the Supplementary Materials. For each subtask, we uniformly randomized the scripted grasping poses by 1 cm in each translation dimension to introduce variations to the policy. This task was considered

successful if all the panels were assembled successfully, and each subpolicy was allowed a maximum of two attempts. For this task, we performed 10 trials for the chained policies, because it is a very long-horizon task.

Car dashboard assembly

As illustrated in Fig. 2, the car dashboard assembly task involved two stages: Both of the arms first needed to grasp on appropriate locations of the workpiece, lift it up, and assemble it onto the dashboard. The workpiece was randomized in a 3 cm–by–3 cm region with 10° in rotation. The task was considered successful if all the pins of the workpiece were fully inserted into the corresponding holes on the dashboard. This task requires precise manipulation and bimanual coordination: The two arms must coordinate the timing of the motions and gripper closure to lift the workpiece up, rotate it, and align multiple pins at the same time.

Object handover

In this task, two robot arms were required to coordinate transfer of an object from one basket to the other. The right arm first picked up the object from a basket on the right side. Then, it handed the object over to the left arm, which placed it precisely into a basket on the left side. The task was considered successful if the object was placed flat on the right basket. The handover part is challenging because the robots' grippers must coordinate the timing of the motions to prevent the object from falling down.

Timing belt assembly

In this task, two robot arms collaborated to assemble a timing belt onto pulleys and actuate the tensioner. This task is part of the National Institute of Standards and Technology board assembly challenge (*69*). The process involved locating and manipulating the randomly placed belt, coordinating precise motions to thread the belt over two pulleys, and simultaneously actuating the tensioner to accommodate the belt. Success was achieved when the belt was properly threaded onto both pulleys and the tensioner was securely tightened. This task presents several challenges. The belt can deform unpredictably during the assembly process, requiring adaptive manipulation. The arms must coordinate precisely, with carefully timed movements, to thread the belt effectively. The timing of tensioner adjustments is critical, because the tensioner must allow the belt to be threaded without jamming. Throughout the process, the policy must continuously adjust to the changing state of the flexible belt and the overall system configuration. The complexity of this task stems from the need to handle a deformable object while precisely coordinating bimanual actions and managing the tensioner mechanism. This requires the policy to develop sophisticated, reactive behaviors to succeed consistently.

Jenga whipping

In this task, the robot was supposed to whip out a specific block from a Jenga tower without toppling over the tower. The nature of this task was largely different from that of the previous tasks, in that it required the robot to learn a highly dynamic open-loop behavior, as opposed to the reactive closed-loop behavior required in the previous tasks. The dynamics of this task are intractably complex: The deformable whip travels at a very high speed and interacts with the surrounding compressed air, making its motion difficult to predict. In addition, determining the precise force needed to remove a specific block without destabilizing the entire tower introduces further complexity because of the intricate contact dynamics involved. It is imperative for the policy to develop a reflex-like behavior by observing the outcomes of its own actions, intuitive physics, and the interactions between the whip and the blocks. This allows the robot to execute precise and consistent motions to successfully remove the target block without causing the tower to collapse. Note that for this specific task, we initialized an offline dataset with 30 expert demonstrations rather than using real-time human corrections. This was chosen deliberately, because incorporating human feedback during training would be both impractical and unsuitable given the task's unique characteristics.

Object flipping

In this task, an object was randomly placed on a pan attached to the robot's end effector, and the robot was tasked to flip the object over a horizontal axis. The task was considered successful if the object was flipped to the opposite side and remained in the pan. Because the initial placement of the object was randomized, the policy must learn to adapt to these variations, e.g., moving the object to a more favorable position before executing the flip motion. The task is similar to the Jenga task, requiring precise and sophisticated open-loop behaviors. However, it also involves a closed-loop component, because the policy might need to reposition the object initially.

Performance results of HIL-SERL

In this subsection, we present the experimental results for all of the tasks mentioned above. For each task, we report the success rate, cycle time, and training time. The training time includes all scripted motion, policy rollouts, intended stops, and onboard computation, which was carried out on a single NVIDIA RTX 4090 graphics card. During both training and evaluation, we randomized the initial states as detailed in the task description above. Unless otherwise noted, all results are based on 100 evaluation trials. Our evaluation protocol can be found in the Supplementary Materials.

A central claim of this paper is that HIL-SERL outperforms IL methods based on human teleoperation. To substantiate this, it is crucial to compare relevant IL methods fairly under equivalent settings. Naïve IL approaches often suffer from error compounding issues, as noted in (*70*). DAgger and its variants (*70*, *71*) address this problem by incorporating human corrections to refine the policy through supervised learning. Our method also leverages human corrections, but it instead uses them to optimize the policy through RL based on task-specific rewards. Therefore, we compared our approach with IL by training a baseline with Human-Gated DAgger (HG-DAgger) (*71*), using the same number of human demonstrations and corrections as that used in RL. We first pretrained a base policy with behavioral cloning (BC) using a number of offline human demonstrations equivalent to that provided to our method. We then ran this policy and collected human expert corrections, such that the total number of trials and interventions matches that of RL training. Specifically, we ran it for the same number of episodes as our method and aimed to provide a comparable number of interventions per episode.

This comparison was performed for all tasks except Jenga whipping and object flipping, where interventions were challenging and undesirable. For these tasks, we instead collected 50 and 200 offline demonstrations and trained BC policies as baselines, which took around the same amount of time as training HIL-SERL. This provided a significantly larger number of demonstrations than our method, which typically requires only 20 to 30 demonstrations.

In all our experiments, we used success rates and cycle time as primary metrics to compare different methods. To further validate the effectiveness of our approach, we also report the intervention rate

over time, demonstrating that our policy improved progressively, reducing the need for interventions. Ideally, the intervention rate trends toward zero, indicating that the policy performs autonomously. The experiment results can be found in Fig. 3 and Tables 1 and 2.

First, as shown in Table 1, HIL-SERL achieved a success rate of 100% within 1 to 2.5 hours of real-world training on nearly all the tasks. This is a substantial improvement over the HG-DAgger baseline, which achieved an average success rate of 49.7% across all tasks. The performance gap is more pronounced for the tasks that require more complex behaviors—Jenga whipping, RAM stick insertion, and timing belt assembly.

We also report the number of human interventions over time for nearly all of the tasks in Fig. 3. Specifically, we report the intervention rate, for which we calculated the ratio of intervened time steps to total time steps within an episode and report a running average over 20 episodes. As shown in the figure, the intervention rate decreased as the training progressed, indicating that the policies were improving and were less reliant on human corrections. In addition, we observed that the total duration of interventions decreased markedly. Initially, we issued long, sparse interventions when the policy was immature. As the policy improved, shorter interventions were sufficient to correct fewer mistakes. In contrast, the HG-DAgger policies required more frequent interventions to correct the policy, and the total duration of interventions did not necessarily decrease over time. Thus, our method attained better performance with less human supervision.

Our method outperforms HG-DAgger because of key advantages of RL. RL explores a wider range of states and directly optimizes task-specific rewards, whereas DAgger's reliance on human corrections can introduce inconsistencies and limit state exploration. Because RL learns from its own state distribution and corrects errors autonomously, it overcomes the constraints of human demonstrations, resulting in more robust policies. These empirical findings align with theoretical results discussed in (*50*), which demonstrate that RL policies can, in principle, outperform DAgger. The performance gap tends to widen as the suboptimality of human corrections increases, a scenario that is more likely to occur as tasks become more complex.

Another important aspect to consider is the cycle time or the time it takes to complete the task. On average, the HG-DAgger policies achieved an average cycle time of 9.6 s, whereas our method achieved an average cycle time of 5.4 s. This indicates an improvement of 1.8 times. This improvement is expected, because IL methods lack mechanisms to deal with suboptimality in human demonstrations. In contrast, RL can leverage dynamic programming to optimize for the discounted sum of rewards. For a discount factor $\gamma < 1$, this approach encourages the policy to acquire rewards faster, resulting in shorter cycle times compared with those achieved by imitating human demonstrations.

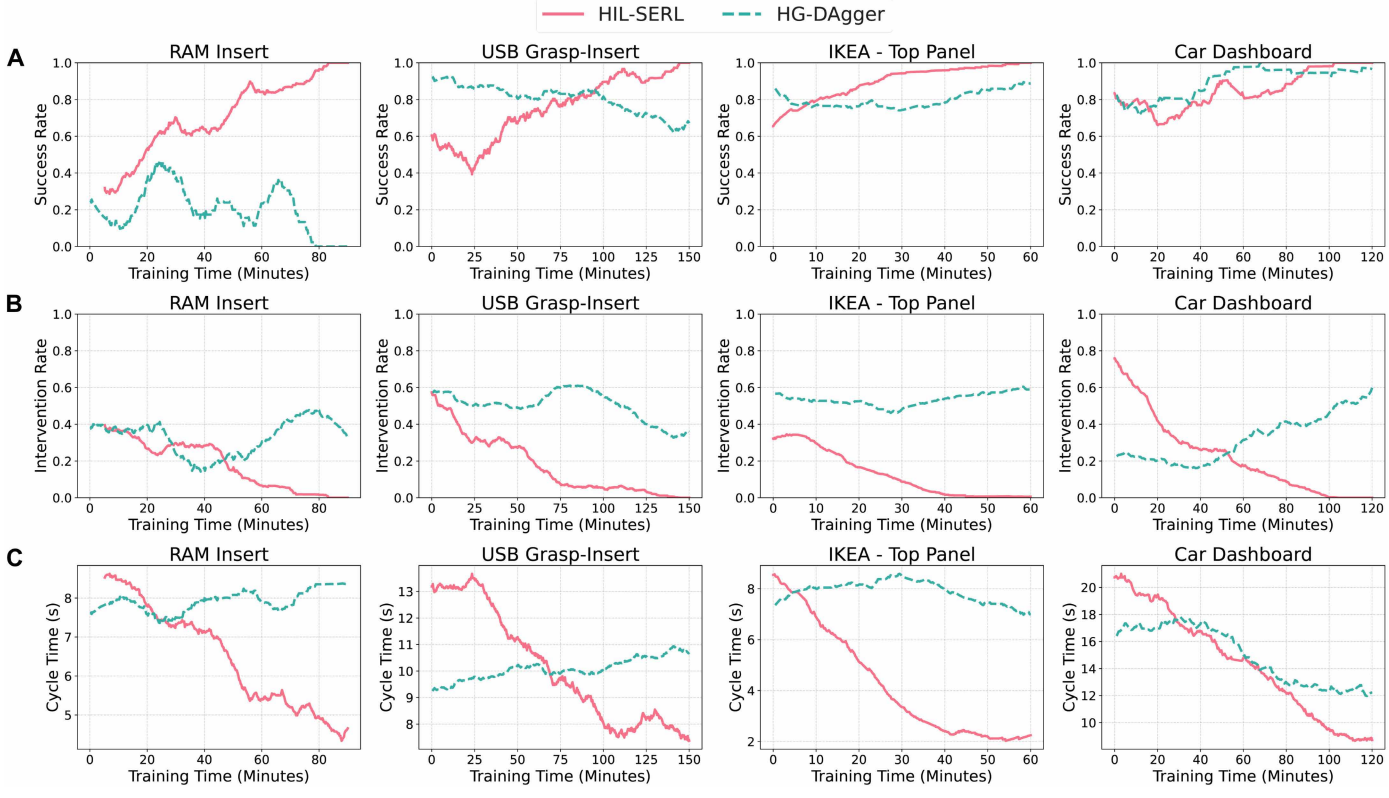


Fig. 3. Learning curves for experimental tasks. The figure presents (**A**) the success rates, (**B**) the cycle times, and (**C**) the intervention rates for both HIL-SERL and DAgger across a few representative tasks, displayed as a running average over 20 episodes. For HIL-SERL, the success rate increased rapidly throughout training, eventually reaching 100%, whereas the intervention rate and cycle time progressively decreased, with the intervention rate ultimately reaching 0%. For HG-DAgger, the success rate fluctuated throughout training episodes and did not necessarily increase as training progressed. Because interventions occurred frequently, leading to successful outcomes, the true policy success rate is likely lower than the curve suggests. In addition, the intervention rate did not consistently decrease over time, indicating that the policy was not steadily improving. This is reflected in the cycle time as well, which showed no improvement, given that DAgger lacks a mechanism to enhance performance beyond the provided training data. Additional plots are available in the Supplementary Materials.

Table 1. Comparison of HIL-SERL against IL baselines on all tasks in terms of success rate and cycle time. Comparison of IL and RL success rates and cycle times for various tasks. All metrics were reported over 100 trials per task, except for the IKEA whole assembly task, which involved 10 trials. For all tasks except Jenga whipping and object flipping, we trained the IL baselines with HG-DAgger, whereas those two tasks were trained with naive BC. The training time reflects the wall clock time required for HIL-SERL policies to converge and the amount of time we spent collecting intervention data and demos for IL methods. —, not applicable.

		Success rate (%)		Cycle time (s)	
Task	**Training time (hours)**	**IL**	**HIL-SERL (ours)**	**IL**	**HIL-SERL (ours)**
RAM insertion	1.5	29	100 (+245%)	8.3	4.8 (1.7× faster)
SSD assembly	1	79	100 (+27%)	6.7	3.3 (2× faster)
USB grasp-insertion	2.5	26	100 (+285%)	13.4	6.7 (2× faster)
Cable clipping	1.25	95	100 (+5%)	7.2	4.2 (1.7× faster)
IKEA—side panel 1	2	77	100 (+30%)	6.5	2.7 (2.4× faster)
IKEA—side panel 2	1.75	79	100 (+27%)	5	2.4 (2.1× faster)
IKEA—top panel	1	35	100 (+186%)	8.9	2.4 (3.7× faster)
IKEA—whole assembly	—	10	100 (+900%)	—	—
Car dashboard assembly	2	41	100 (+144%)	20.3	8.8 (2.3× faster)
Object handover	2.5	79	100 (+27%)	16.1	13.6 (1.2× faster)
Timing belt assembly	6	2	100 (+4900%)	9.1	7.2 (1.3× faster)
Jenga whipping	1.25	8	100 (+1150%)	—	—
Object flipping	1	46	100 (+117%)	3.9	3.8 (1.03× faster)
Average	—	49.7	100 (+101%)	9.6	5.4 (1.8× faster)

Table 2. Comparison of success rate of HIL-SERL against various other baselines on select tasks. DP and BC were trained with 200 demonstrations, whereas HG-DAgger was trained with the same number of episodes and interventions as RL. IBRL, residual RL, and DAPG were initialized with 200 demonstrations. Our method was also ablated with two versions: one initialized from scratch without demonstrations or interventions (HIL-SERL no demo no itv) and another initialized from demonstrations but without corrections (HIL-SERL no itv).

	RAM insertion	**Dashboard assembly**	**Object flipping**	**Average**
DP	27	18	56	34
HG-DAgger	29	41	46	39
BC	12	35	46	31
IBRL	75	0	95	57
Residual RL	0	0	97	32
DAPG	8	18	72	33
HIL-SERL no demo no itv	0	0	0	0
HIL-SERL no itv	48	0	100	49
HIL-SERL (ours)	100	100	100	100

Of these experiments, we would note that our method proved to be general and effective across tasks with vastly different physical properties, generating both open-loop and closed-loop policies well suited for each task's specific requirements. For precise manipulation tasks, such as assembling a timing belt or inserting a RAM stick, the policy learned to associate task-relevant visual features with appropriate twist motions. It performed continuous visual servoing behavior, reacting to streaming observations in real time and adjusting its motion until reaching the target. In contrast, for tasks like Jenga whipping and object flipping, the policy learned to predict potential outcomes of its actions through interaction. It then precisely refined its motion to achieve the desired outcome while maintaining consistency in execution. We also offer an in-depth analysis of the learned behavior, which we defer to a later section.

Overall, our experiments show that our method is both general and effective. It successfully learned policies for a variety of challenging manipulation tasks using the same approach, achieving high performance across all tasks. In addition, it achieved such performance within practical training times, even for high-dimensional observations and action spaces, such as those required for bimanual manipulation.

Robustness results

To test the zero-shot robustness of policies learned by our method, we provide a set of qualitative results in Fig. 4 and movies S1 to S5.

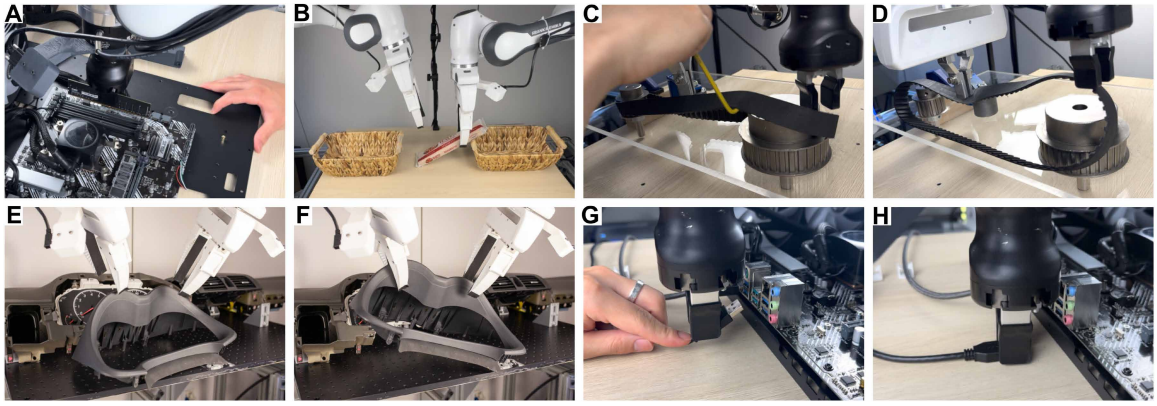


Fig. 4. Robustness evaluation for policies learned by our method. (**A**) RAM insertion under external perturbations, such as a moving motherboard. (**B**) Retrying behavior during a handover task after the grippers are forced open. (**C** and **D**) Reactive responses in the timing belt task, addressing both external disturbances and unexpected deformations during executions. (**E** and **F**) In the dashboard assembly task, the policy performs regrasps after one or both grippers are forcibly opened. (**G** and **H**) In the USB grasp-insertion task, the policy adapts to external disturbances and poor grasps by releasing and regrasping the object.

These results demonstrate the policy's ability to adapt dynamically to variations on the fly and handle external disturbances, such as objects being intentionally dropped by a human from the gripper, or cases where a human deliberately forced the gripper open during task execution.

In the timing belt assembly task, the belt can undergo arbitrary deformation, and the policy was supposed to adapt to these variations during the assembly process. In addition, we introduced external disturbances to the belt to further test the robustness of the policy. These disturbances include artificially perturbing the belt's shape or repositioning it dynamically during the assembly process, as in Fig. 4 (C and D). In the RAM insertion task, the learned policy successfully inserted the RAM stick even when the motherboard was moved to any position within a 10 cm–by–10 cm region with up to 45° of rotation during evaluation, thanks to the egocentric representation of the proprioceptive observation, as illustrated in Fig. 4A. For the car dashboard assembly and object handover tasks, after the policy grasped the object, we forced the gripper to open during task execution. The policy reacted to this unexpected situation by retrying to grasp the object and proceeding with the rest of the tasks, as presented in Fig. 4 (B, E, and F). In the USB connector grasp-insertion task, we varied the initial pose of the USB connector and occasionally forced it out of the gripper to simulate poor grasp poses. The policy adapted by releasing the connector and regrasping it to achieve a better pose for insertion, as seen in Fig. 4 (G and H). These robust behaviors are achieved through autonomous exploration during the RL training phase. For example, the policy learned to associate the grasping pose with the downstream insertion task and regrasp the object if necessary. However, these robust behaviors are usually hard to achieve with IL methods, because they lack this mechanism to autonomously explore and learn from the outcomes of their actions.

Additional baseline comparisons

To validate the effectiveness of design choices in our method, we conducted additional comparisons on three representative tasks: car dashboard panel assembly (dual-arm coordination), RAM insertion (precise manipulation), and object flipping (dynamic manipulation). We compared our approach against several state-of-the-art methods to highlight different aspects of its performance. To illustrate the effect of human interventions, we performed ablation studies varying the number of human demonstrations and corrections. To showcase how effectively our method incorporates and leverages human demonstrations, we compared it against residual RL (*44*), demonstration augmented policy gradient (DAPG) (*46*), and imitation bootstrapped reinforcement learning (IBRL) (*45*). In addition, we compared it against diffusion policy (DP) (*72*) to ensure that the task difficulty did not stem solely from multimodality in human demonstrations. These comprehensive comparisons validate our method's effectiveness and capabilities across diverse manipulation scenarios. The results are presented in Table 2.

RL from scratch, without any demonstrations or corrections, achieved a 0% success rate on all tasks because the policy was unable to receive any sparse reward signals from random exploration. To validate the importance of online human corrections, we increased the number of demonstrations in the offline buffer of SERL 10-fold, from the usual 20 to 200. However, without any online corrections, this approach resulted in significantly lower success rates compared with those of HIL-SERL, including a complete failure (0% success) on complex tasks such as the car dashboard assembly. This result confirms the crucial role of online corrections in facilitating policy learning. These results confirmed the crucial role of both offline demonstrations and on-policy human interventions in guiding policy learning, especially for complex manipulation tasks that require continuous reactive behaviors.

For the object flipping task, we trained BC policies using both 20 and 200 demonstrations. The results from these two BC policies were quite similar, with success rates of 47 and 46%, respectively, although the number of demonstrations increased 10-fold. We observed that the learned policy often failed to generate the correct motion and velocity required to toss the object into the air. This result indicates that merely imitating human demonstrations is insufficient to solve this task, although it is largely open-loop.

Another important aspect to consider is how our method handles demonstrations compared with others. To compare against the mentioned baselines, we collected 200 demonstrations for each task. This number is substantially larger than the number of offline demonstrations in our method, which is usually around 20 to 30 in the offline replay buffer. For residual RL and IBRL, we trained BC

policies with these demonstrations to feed into their algorithm pipeline. For DAPG, we stored these 200 demonstrations in a separate buffer and regularized the policy actions toward them. Overall, our method consistently outperformed these baselines by large margins, as shown in Table 2.

This can be interpreted as follows. Residual RL depended on a pretrained BC base policy to facilitate the learning process. However, this approach can be problematic for tasks that require precise manipulation, such as car dashboard assembly or RAM insertion. In these scenarios, IL methods, including BC, often performed inadequately. As a result, the RL policy learning process can experience substantial failures. For IBRL, the actor is a hybrid of a BC policy and an RL policy, making the behavior more "BC-like." This approach struggles on tasks where BC performs poorly. For DAPG, because it directly regularized the policy actions toward the demonstrations, it is expected that the policy performed similarly to the BC policies. Therefore, it underperformed our method on tasks that require more reactive and complex behaviors.

The effectiveness of our method came from the off-policy nature of the underlying RL algorithm, which dynamically weighted human data on the basis of its relevance to the current policy optimization objective. Unlike (*44*–*46*), which heavily relied on the quality of human demonstrations, our method has a mechanism that allows for efficient use of human data early in training while enabling the agent to progressively surpass human-level performance. Crucially, it prevented the agent from being constrained by the limitations of human demonstrations, striking a balance between bootstrapping from demonstrations and discovering novel, superior strategies through autonomous exploration.

To compare with the DP (*72*), we trained the policies using 200 demonstrations for each task, which is substantially more than the 20 demonstrations available in the offline replay buffers used by our method. We report the results using the experimentally optimal algorithm parameters, such as observation and action chunking length, and the length of the applied action sequence in the Supplementary Materials. On the RAM insertion and car dashboard panel tasks, DPs achieved success rates of 27 and 28%, respectively. On the object flipping task, the success rate was 56%. This performance was significantly lower than that of our method and even falls short of the HG-DAgger baseline. This outcome is not unexpected, given that the primary strength of DPs is in learning a more expressive policy distribution to accurately "memorize" robot motions. However, these tasks required more "closed-loop" reactive behaviors, such as continuous visual servoing to correct motion errors. Therefore, the advantage of DPs in learning an expressive policy distribution does not necessarily lead to better performance in these tasks.

Analysis

To provide deeper insights into our results, we offer a detailed analysis of the learned policies. This analysis focused on two key aspects: reliability and learned behaviors. We examined why the learned policies consistently achieved high success rates across diverse tasks, investigating the factors that contributed to their robustness. In addition, we delved into the nature of the behaviors acquired by the policies, particularly exploring the distinction between reactive and predictive strategies. This comprehensive analysis aimed to shed light on the underlying mechanisms that contribute to the effectiveness of our approach in solving complex manipulation tasks.

Reliability of the learned policies

One key aspect of HIL-SERL's performance is its high reliability, achieving a 100% success rate across all tasks. We argue that this reliability comes from RL's inherent ability to self-correct through policy sampling, allowing the agent to continuously improve by learning from both successes and failures. In contrast, IL approaches, including interactive methods, lack this self-correction mechanism, making it substantially more challenging to achieve comparable performance with the same amount of data. Although there is existing theoretical work on the convergence of Q-learning (*73*–*76*), our analysis focused on providing an intuitive understanding of the training dynamics.

To illustrate this, we analyzed the RAM insertion task, which requires precise manipulation and is easily visualized because of symmetrical randomization in the X and Y directions. We plotted heatmaps of state visitation counts across time steps for different policy checkpoints in Fig. 5 on the basis of the end effector's y and z positions. Through policy learning, we observed the gradual development of a funnel-like shape connecting the initial states to the target location. This funnel became more defined as empty regions were filled and narrowed when approaching the target, indicating increased policy confidence and precision. Over time, the mass concentrated in areas likely to succeed. We then introduced the concept of "critical states," defined as states where the Q-function variance is large. We computed this variance using

$$\mathrm{Var}[Q(\mathbf{s},\mathbf{a})] = \mathbb{E}_{\varepsilon\sim\mathcal{U}[-c,c]}\left[\left(Q(\mathbf{s},\mathbf{a}+\varepsilon) - \mathbb{E}_{\varepsilon\sim\mathcal{U}[-c,c]}(Q(\mathbf{s},\mathbf{a}+\varepsilon))\right)^2\right] \quad (1)$$

For each data point and its associated policy checkpoint, we added uniform random noise from [−0.2, 0.2] to the action (normalized to [−1, 1]) at every state and computed the Q-function variance using Monte Carlo sampling with 100 samples. A large variance indicates that the state was critical to the policy's success, because taking a different action would result in significantly different (usually much smaller) Q values. Figure 5 also shows heatmaps of Q values and their variances at different states. These plots demonstrate the policy developing a funnel where the most visited states gained higher Q values and higher Q value variances. This indicates that the policy was robustifying the region, effectively connecting important states with actions leading to high Q values through dynamic programming.

In contrast, the heatmap of state visitation counts for HG-DAgger on the same task (fourth row of Fig. 5) shows a much sparser distribution. The funnel shape is less distinct, and the mass is more spread out, with states visited more uniformly compared with the RL case. This is because RL can explore autonomously and use dynamic programming directed by task rewards, whereas DAgger can only explore around the current policy. Consequently, to achieve similar performance, DAgger may require substantially more demonstrations and corrections, as well as careful attention from the human operator to ensure data quality.

This type of stabilizing behavior within a funnel has been developed in state-based dexterous manipulation and motion planning (*77*, *78*). However, our approach differs in that we directly leveraged perceptual inputs and used RL exploration to autonomously form the funnel. An analogous concept in optimal control is the development of controllers that stabilize around nominal trajectories using local feedback (*24*). In our case, demonstrations and corrections can be regarded as "nominal trajectories" around which RL methods develop funnels for stabilization.

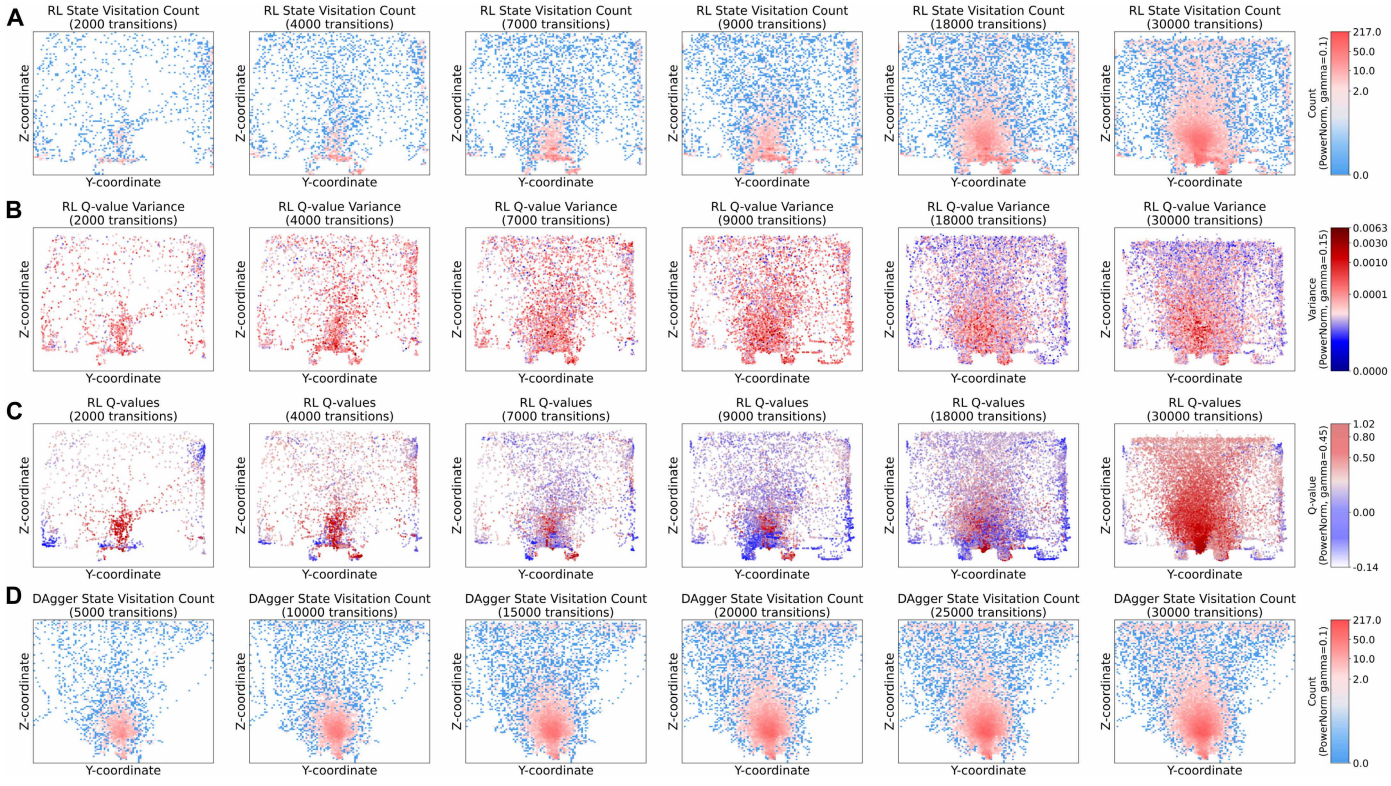


Fig. 5. Visualization of policy training dynamics. (**A**) State visitation heatmaps during HIL-SERL training: The policy progressively forms a funnel shape, concentrating more on areas around the demonstrations and corrections, showing robustification in these regions. (**B**) *Q* value variance scatterplots throughout training: States within the funnel show increased *Q* value variance, indicating that the policy was gaining greater confidence in actions that led to successful outcomes. (**C**) *Q* value scatterplots across training: Critical states, marked by higher *Q* value variance, were also associated with high *Q* values. (**D**) State visitation heatmaps during HG-DAgger training: The funnel shape is less pronounced and shows more diffused distribution of visitation density.

Reactive policy and predictive policy

To solve most of the high-precision manipulation tasks, we need a closed-loop reactive policy that responds rapidly to immediate sensory feedback, therefore enabling precise adjustments in real time. On the other hand, for the dynamic manipulation tasks, such as Jenga whipping and object flipping, it is desirable to use an open-loop predictive policy that plans ahead and execute the motion consistently. To see this, we picked two representative tasks requiring these two different types of policies—Jenga whipping and RAM insertion—for analysis. To visualize the differences between these policy types, we plotted the computed actions from the trained Gaussian policies for both tasks in Fig. 6.

For both tasks, we analyzed three successful trajectories by plotting the policies' computed SD and mean over time. From these plots, we observed that although the mean actions cover a wide range of values in both cases, the SDs reveal distinct policy behaviors. In the Jenga whipping task, the SD remained consistently low (very close to 0) across time steps. This indicates a highly confident and consistent policy, ideal for tasks where open-loop behaviors are desirable. Similar to a tennis player developing a reflex, the policy learned to execute a precise, preplanned motion. Through environmental interactions, it refined this motion to minimize prediction errors, resulting in consistent execution. Conversely, the RAM insertion task exhibits a different pattern. Initially, the SD was much higher (around 0.6), reflecting uncertainty when approaching the target early on. However, it decreases rapidly over time, suggesting an initially coarse approaching motion that became more precise when near the target. This task demands a reactive policy capable of error correction in various scenarios, given that predictive control over a long horizon is impractical because of the task's precision requirements. This reactive behavior is even more pronounced in complex manipulation tasks such as dashboard panel assembly or timing belt installation. In these cases, the policy must continuously adjust its actions on the basis of sensory feedback, often requiring multiple attempts to achieve success, such as breaking contact and reapproaching the target, as illustrated in Fig. 6. The higher variance in these scenarios indicates the policy's readiness to react swiftly to changing conditions.

It is worthwhile to note that this type of reactive behavior is acquired by the agent through interaction with the environment. In other words, the agent develops this behavior "for free"—we do not explicitly formulate the problem to solve for a specific dynamic behavior. Instead, the desired response emerges naturally as part of the solution through ongoing interactions. Previous work (*79*–*81*) has attempted to formulate these contact-rich manipulation problems as mixed-integer programming for the resulting hybrid systems, which allows the policy to plan different modes of contact and the accommodating motions. However, these methods can quickly become computationally intractable as the planning horizon increases, because the number of possible contact modes grows exponentially

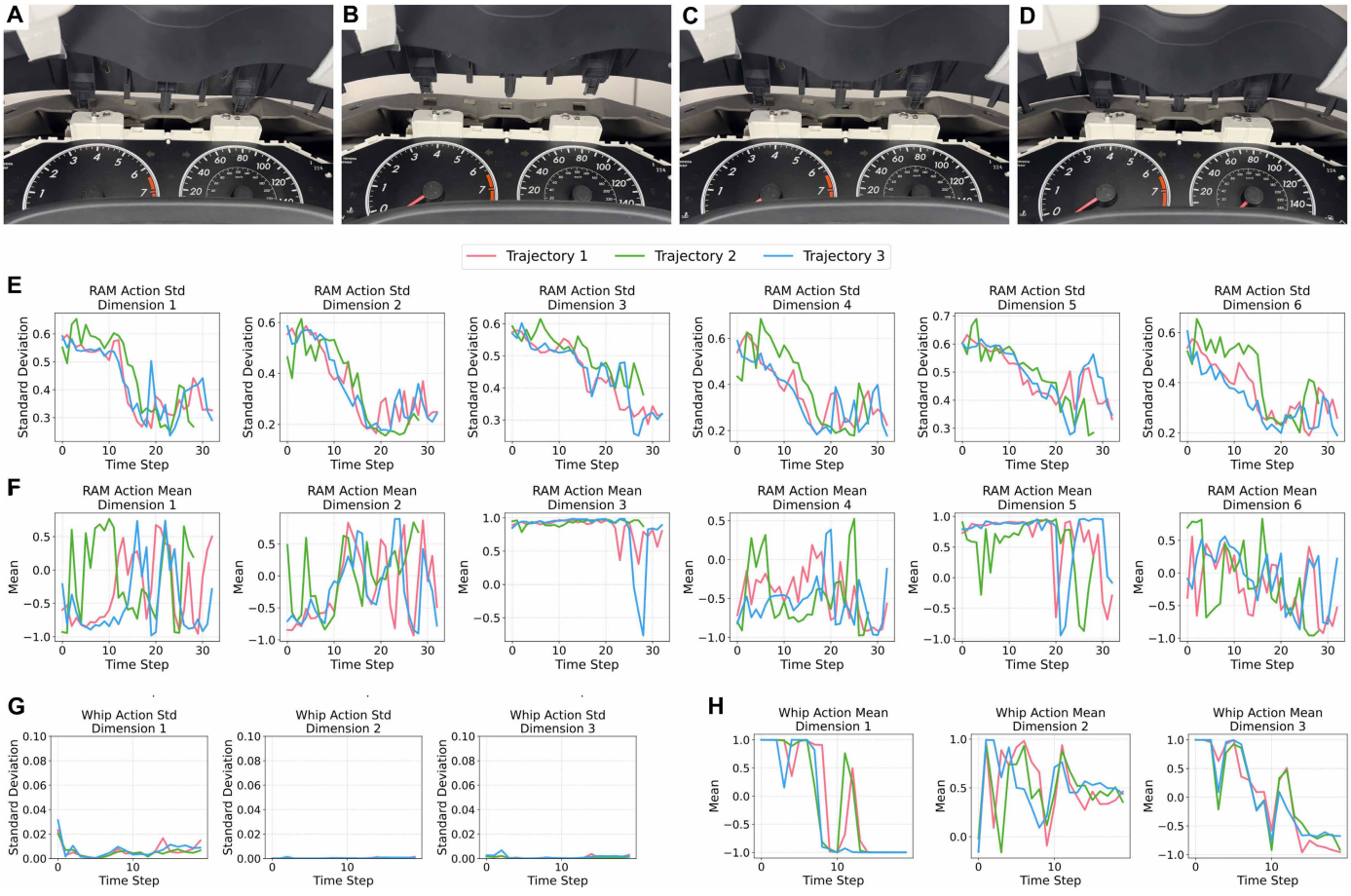


Fig. 6. Reactive versus predictive behavior. (**A** to **D**) A sequence of reactive behaviors in the dashboard assembly task: After getting stuck in contact, the policy broke the contact by quickly lifting two arms, then reestablishing the contact when approaching the target, and lastly succeeding in the insertion. (**E**) Variance plots from trained Gaussian policies in the RAM insertion task, showing three trajectories. Initial variance is high but rapidly decreases as the target is approached. Std, standard deviation. (**F**) Mean plots from trained Gaussian policies in the RAM insertion task, with values ranging from −1 to 1. (**G**) Variance plots in the Jenga whipping task, remaining consistently low (near 0), indicating stable execution and open-loop behavior. (**H**) Mean plots in the Jenga whipping task, with values between −1 and 1, demonstrating consistent behavior across three trajectories.

with the length of the planning horizon. In addition, they require accurate state estimators, which are not always available for many real-world tasks.

In contrast, our method directly leverages perception to learn reactive behaviors upon encountering contact. Through interaction, it encodes essential dynamics required to find a solution, rather than treating these dynamics as part of the problem formulation. Prior approaches, however, incorporate complex or intractable dynamics within the problem formulation itself, making these solutions harder to derive and less scalable.

Overall, our approach demonstrates the flexibility to learn these distinct policy types within a unified algorithmic framework. By interacting with the environment and observing the outcomes of its actions, the method adapts to the specific demands of each task. This adaptability enables the system to effectively address tasks that require diverse behaviors, spanning a wide range of manipulation challenges.

DISCUSSION

The presented results substantially advance the published state of the art in robotic manipulation. Our research demonstrates that with the right design choices, model-free RL can effectively tackle a variety of complex manipulation tasks using perception inputs, directly training in the real world within a practical timeframe. Trained policies from this approach are highly performant subject to a modest amount of variation of initial placements, achieving nearly perfect success rates that substantially exceed those of alternative approaches, such as IL, along with cycle times that are also considerably faster. Although our system does not yet address robotic manipulation in the fully general setting—such as semantic generalization—it demonstrated strong general applicability across a range of challenging manipulation tasks, achieving high performance under specific conditions. We believe that the presented results are both practical and relevant for many industrial manipulation scenarios. In particular, the level of randomization handled by our system falls well within the precision capabilities of modern computer vision (CV) systems. In practice, a modern CV system could be used to guide the robot arm into the appropriate range of initial conditions, after which the learned RL skills can be reliably triggered.

Beyond the results themselves, the approach presented in this work can have broader influence. It can serve as a general framework for acquiring a wide range of manipulation skills with high

performance and adapt to variations. This is particularly valuable in high-mix low-volume manufacturing or "make-to-order" production (*82*–*84*). Such production methods have substantial potential in major industries, such as electronics, semiconductors, automotive, and aerospace, because of these industries' need for shorter product life cycles, customization, agility, and flexibility.

We see a number of opportunities for future work. First, our approach can serve as an effective tool to generate high-quality data to train robot foundation models (*85*–*89*). Given that each task requires a relatively short time to train and that the training process is largely autonomous, this framework can be used to develop a variety of skills. Subsequently, data can be collected by executing the converged policies, which can then be distilled into these generalist models. Second, although the current training time is relatively short, each task still requires training from scratch. We can further reduce this time by pretraining a value function that encapsulates the general manipulation capabilities of solving a range of different tasks with distinct robot embodiments. This pretrained value function can then be quickly fine-tuned to address specific tasks.

We also see some limitations of our approach. Although we successfully address a variety of challenging tasks, it remains uncertain whether this method can be further extended to tasks with substantially longer horizons, where the sample complexity issue becomes more pronounced. However, this challenge might be alleviated by improved pretraining techniques or using methods that automatically segment a long-horizon task into a series of shorter subtasks, such as a vision-language model. We also note that our approach relies on a sparse reward function. Although sparse rewards simplify task specification and remove the need for carefully hand-designed shaping, they pose substantial challenges for exploration, especially in tasks with multiple stages or very narrow success conditions. In our experiments, combining sparse rewards with off-policy RL and human interventions has proven effective. However, for substantially longer or more complex tasks (e.g., involving 10 or more substages), additional techniques such as reward shaping or hierarchical learning may be necessary to ensure efficient learning and reliable performance. We also did not perform extensive randomization in our experiments, nor did we test the method's generalization capability in unstructured environments. The primary focus of this paper is to demonstrate that the approach can be general purpose in acquiring a wide range of manipulation skills with high performance under specific conditions. We believe that the challenge of handling randomization can be effectively addressed through a learning curriculum, for example, by progressively increasing the level of randomization during training, as demonstrated in (*36*). In addition, the generalization issue might be resolved by incorporating vision foundation models that are pretrained on large-scale diverse robotic datasets. We hope that this work will pave the way for the use of RL in solving robotic manipulation problems, achieving high performance and eventually deploying robots into the real world.

MATERIALS AND METHODS

Background and problem statement

Robotic RL tasks can be defined via a Markov decision process (MDP) $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \rho, \mathcal{P}, r, \gamma\}$, where $\mathbf{s} \in \mathcal{S}$ is the state observation (e.g., an image in combination with the robot's proprioceptive state information), $\mathbf{a} \in \mathcal{A}$ is the action (e.g., the desired end-effector twist), $\rho(\mathbf{s}_0)$ is a distribution over initial states, $\mathcal{P}$ is the unknown and potentially stochastic transition probabilities that depend on the system dynamics, and $r: \mathbf{S} \times \mathbf{A} \rightarrow \mathbb{R}$ is the reward function, which encodes the task. An optimal policy π is one that maximizes the cumulative expected value of the reward, i.e., $E\left[\sum_{t=0}^{H} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right]$, where the expectation is taken with respect to the initial state distribution, transition probabilities, and policy π. In practice, the policy $\pi(\mathbf{a}|\mathbf{s})$ is usually modeled as a Gaussian distribution parameterized by a neural network.

To implement RL algorithms for robotic tasks, we must carefully select appropriate state observation spaces $\mathcal{S}$ and action spaces $\mathcal{A}$. This involves choosing the right combination of cameras, proprioceptive states, and corresponding robot low-level controllers. For all of our tasks, we used a sparse reward function. This function makes a binary decision on whether a task is successful using a trained classifier. In this setup, the optimization objective $E\left[\sum_{t=0}^{H} \gamma^t r(\mathbf{s}_t, \mathbf{a}_t)\right]$ aims to maximize the probability of success for each trajectory. Ideally, at convergence, the policy should succeed at every attempt.

Specifically, the core RL algorithm that we built on is RLPD (*11*), which we chose for its sample efficiency and ability to incorporate prior data. At each training step, RLPD samples equally between prior data and on-policy data to form a training batch (*90*). It then updates the parameters of a parametric Q-function $Q_\phi(\mathbf{s}, \mathbf{a})$ and the policy $\pi_\theta(\mathbf{a}|\mathbf{s})$ according to the gradient of their respective loss functions

$$\left[\mathcal{L}_Q(\phi) = E_{\mathbf{s},\mathbf{a},\mathbf{s}'}\left[\left(Q_\phi(\mathbf{s}, \mathbf{a}) - \left(r(\mathbf{s}, \mathbf{a}) + \gamma \mathrm{E}_{\mathbf{a}' \sim \pi_\theta}\left[Q_{\bar{\phi}}(\mathbf{s}', \mathbf{a}')\right]\right)\right)^2\right]\right] \quad (2)$$

$$\mathcal{L}_\pi(\theta) = -E_{\mathbf{s}}\left[E_{\mathbf{a} \sim \pi_\theta(\mathbf{a})}\left[Q_\phi(\mathbf{s}, \mathbf{a})\right] + \alpha \mathcal{H}\left(\pi_\theta(\cdot|\mathbf{s})\right)\right] \quad (3)$$

where $Q_{\bar{\phi}}$ is a target network (*91*) and the actor loss uses entropy regularization with an adaptively adjusted weight, α (*92*). To instantiate RLPD, we chose SAC [Soft Actor-Critic (*93*)] as the base RL algorithm, although, in principle, we could use any sample-efficient off-policy RL algorithms.

System overview

Our system is composed of three major components: the actor process, the learner process, and the replay buffer residing inside the learner process, all operating in a distributed fashion, as illustrated in Fig. 7. The actor process interacts with the environment by executing the current policy on the robot and sends the data back to the replay buffer. The environment is designed to be modular, allowing for flexible configuration of various devices. This includes support for multiple cameras, integration of input devices like SpaceMouse for teleoperation, and the ability to control a variable number of robot arms with different types of controllers. An implemented reward function is required to assess the success of a task, which is trained offline using human demonstrations. Inside the actor process, a human can intervene with the robot using a SpaceMouse, which will then take over the control of the robot from the RL policy. We used two replay buffers, one to store offline human demonstrations, called the demo buffer, usually in the range of 20 to 30 trajectories; the other, called the RL buffer, stored the on-policy data.

The learner process samples data equally from the demo and RL replay buffers, optimizes the policy using RLPD, and periodically sends the updated policy to the actor process. In the remainder of

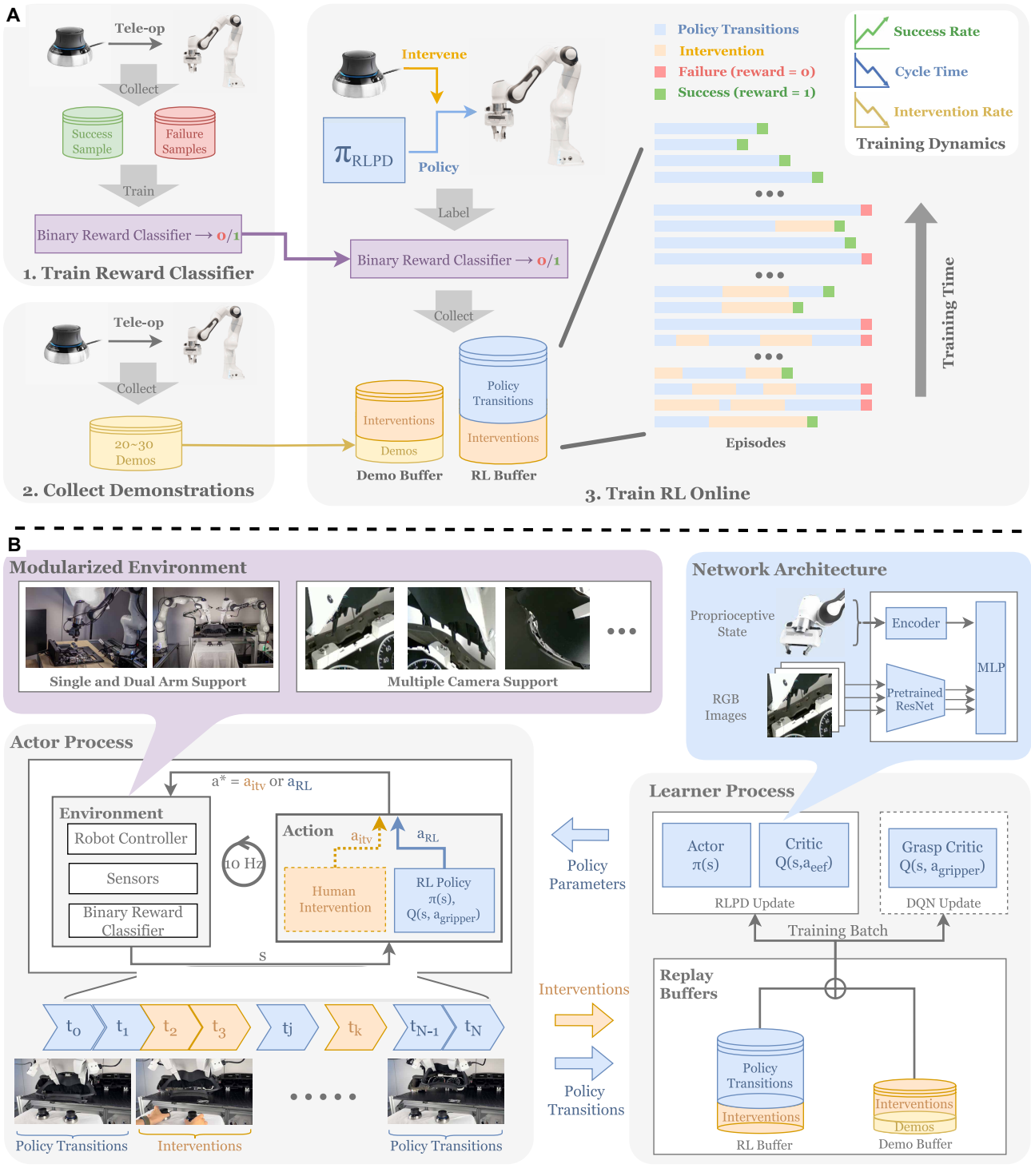


Fig. 7. Training process and overview of HIL-SERL. (**A**) The top diagram illustrates the process for training HIL-SERL. First, we teleoperated the robot to collect positive and negative samples and trained a binary reward classifier. We then collected a small set of demonstrations, which was added to the demo buffer at the start of RL training. During online training, we used the binary classifier as a sparse reward signal and provided human interventions. Initially, we provided more frequent interventions to demonstrate ways of solving the task from various states. We gradually reduced the amount of interventions as the policy reached higher success rates and faster cycle times. (**B**) Overview of HIL-SERL. The main diagram illustrates the architecture of HIL-SERL, which comprises three primary components: the actor process, the learner process, and replay buffers. These components communicate asynchronously to facilitate efficient data flow. The actor process receives updated policy parameters from the learner process, interacts with the environment, and sends collected interaction data to the replay buffers. The environment is modular, supporting various external devices and multiple robotic arms. A human operator can intervene via teleoperation tools, such as a SpaceMouse. The learner process samples data evenly from two replay buffers and updates the policy using RLPD. When gripper control is required, a grasp critic is additionally trained with DQN.

this section, we provide details about the design choices we made for each component.

System design choices

The sample efficiency of the proposed system is crucial, because each minute of training to acquire data incurs a cost. Therefore, the training time must remain within a practical range, particularly when handling high-dimensional inputs like images. In addition, the downstream robotic system must accommodate the RL policy so as to ensure a smooth and efficient learning process. For example, the actual low-level robot controller would require great care, particularly for most of the precise contact-rich tasks where the robot physically interacts with objects in the environment. Not only does this controller need to be accurate, but it also must be safe enough that the RL algorithm can explore with random actions during training. Thus, to develop a system capable of performing sample-efficient policy learning in the real world, we made the following design choices.

Pretrained visual backbones

To facilitate the efficiency of the training process, we used pretrained vision backbones to process image data. Whereas this approach is now a common practice in CV for the purpose of robustness and generalization (*94*–*96*), in RL, this treatment offers additional benefits, such as optimization stability and exploration efficiency (*76*, *97*), making it particularly advantageous for real-world robotic RL training. Our neural network architecture, illustrated in Fig. 7, processes multiple images from cameras using the same pretrained vision backbone. Specifically, we used a ResNet-10 model (*98*), pretrained on ImageNet (*99*), to generate output embeddings. These embeddings are then concatenated and further integrated with processed proprioceptive information, facilitating a more efficient and effective learning process.

Reward function

One crucial aspect of an RL system is the reward function, which is used to guide the learning process and determine the quality of the policy. Although there are prior works using reward shaping to accelerate the learning process (*100*–*102*), this procedure tends to be task specific and time consuming to design. In some complex tasks, it is simply not feasible to perform such reward shaping. We found that using a sparse reward function, alongside human demonstrations and corrections, offered a straightforward and effective setup for a variety of tasks. Specifically, we collected offline data and trained a binary classifier for each task, which only granted a positive reward upon task completion and zero otherwise.

Downstream robotic system

To accommodate the policy learning process, we made a few particularly important design choices to the downstream robotic system. To facilitate spatial generalization, we represented the robot's proprioceptive state in a relative coordinate system, allowing for an egocentric formulation. Essentially, at the beginning of each training episode, the pose of the robot's end effector was randomized uniformly within a predefined area in the workspace. The robot's proprioceptive information was expressed with respect to the frame of the end effector's initial pose; the action output from the policy was relative to the current end-effector frame. This procedure simulated physically moving the target when viewed relatively from the frame attached to the end effector. As a result, the policy could succeed even if the object moved or, as in some of our experiments, was perturbed in the middle of the episode. For tasks involving dealing with contact, we used an impedance controller with reference limiting in the real-time layer to ensure safety as in (*48*). For dynamic tasks, we directly commanded feedforward wrenches in the end-effector frame to accelerate the robot arm. Although it did not perform closed-loop control around acceleration, this simple open-loop control was sufficient for the considered tasks. For details regarding the observation representation and the controller design, please refer to the Supplementary Materials.

Gripper control

For tasks involving the control of grippers, we used a separate critic network to evaluate discrete grasping actions. Although this approach might initially seem like an additional overhead or somewhat unconventional, it has proven to be highly effective in practice, particularly when combined with human demonstrations and corrections. The discrete nature of gripper actions in our tasks made approximating them with continuous distributions more challenging, particularly in the complex tasks considered in this paper. By using discrete actions, we simplified the training process and improved the overall effectiveness of the RL system. Specifically, we solved two separate MDPs in these tasks, $M_1 = \{S, \mathcal{A}_1, \rho_1, \mathcal{P}_1, r, \gamma\}$ and $M_2 = \{S, \mathcal{A}_2, \rho_2, \mathcal{P}_2, r, \gamma\}$, where $\mathcal{A}_1$ and $\mathcal{A}_2$ are the continuous and discrete action spaces, respectively. They both take in the same state observations from the environment such as images, proprioception, gripper status, and so forth. The discrete action space $\mathcal{A}_2$ consists of all possible discrete actions. For a single gripper, these actions are "open," "close," and "stay." If two grippers are involved, then the action space expands to $3^2 = 9$ combinations, accounting for all possible actions each gripper can take. The critic update for $\mathcal{M}_2$ follows the standard Deep Q Network (DQN) practice (*91*) with an additional target network to stabilize training as follows

$$\mathcal{L}(\theta) = \mathbb{E}_{\mathbf{s},\mathbf{a},\mathbf{s}'}\left[\left(r + \gamma Q_{\theta'}\left(\mathbf{s}', \underset{\mathbf{a}'}{\operatorname{argmax}} Q_{\theta}(\mathbf{s}', \mathbf{a}')\right) - Q_{\theta}(\mathbf{s}, \mathbf{a})\right)^2\right] \quad (4)$$

where θ' is the target network, which can be obtained by Polyak averaging with current network parameters (*103*). At training or inference time, we first query the continuous actions from the policy in $\mathcal{M}_1$ and then query the discrete actions from the critic in $\mathcal{M}_2$ by taking the argmax over the critic's output; we then apply the concatenated actions to the robot.

Human-in-the-loop RL

With the system-level design choices in place, we now describe the human-in-the-loop procedure that we used to accelerate the learning process. It is well established from the RL theory literature that the sample complexity of learning an optimal policy is closely tied to the cardinality of the state and action spaces as well as the task horizon (*75*, *104*–*106*), assuming an appropriate exploration policy. These factors collectively serve as proxies for the "upper bound" on the complexity of tasks that can be feasibly solved. Specifically, increases in the size of the state/action spaces, task horizon, or their combinations lead to a proportional rise in the number of samples required to learn an optimal policy, eventually crossing a threshold where real-world training of RL policies becomes impractical.

To tackle this challenge in real-world robotic RL training, we incorporated human-in-the-loop feedback to guide the learning process to help the policy explore more efficiently. Specifically, a human operator supervises the robot during training and provides corrective actions when necessary, as illustrated in Fig. 7. For an autonomous rollout trajectory from time step t_0 to t_N, a human can

intervene at any time step t_i where $t_0 \leq t_i < t_N$. During an intervention, the human takes control of the robot for up to N steps. Multiple interventions can occur within a single trajectory, as illustrated by the yellow segments in Fig. 7. When a human intervenes, their action a_{itv} is applied to the robot instead of the policy's action a_{RL}. We stored the intervention data in both the demonstration and RL data buffers. However, we added the policy's transitions (i.e., the states and actions before and after the intervention) only to the RL buffer. This approach has proven effective in enhancing policy training efficiency.

This intervention is crucial in scenarios where the policy leads the robot to an unrecoverable or undesirable state or when it becomes stuck in a local optimum that would otherwise require substantial time to overcome without human assistance. This procedure is similar to that of HG-DAgger (*71*), where a human takes over the control of the robot to collect data when the policy is performing poorly; however, our approach uses these data to optimize the policy with RL rather than supervised learning, similar to (*50*). In our setup, the human operator engaged with a SpaceMouse 3D mouse to provide corrective actions to the robot.

In the beginning of the training process, the human intervenes more frequently to provide corrective actions, gradually decreasing the frequency as the policy improves. In our experience, we note that the policy improves faster when the human operator issues specific corrections while letting the robot explore on its own otherwise.

Training process

To articulate the training process of our system and assist readers in reproducing our results, we provide a detailed walkthrough of the steps involved in each of our experiments in this section, visually illustrated in Fig. 7 as well. First, we selected cameras that were most suitable for the task. Wrist cameras are particularly useful for facilitating the spatial generalization of the learned policy because of the egocentric views they provide. However, if wrist cameras alone could not provide a full view of the environment, then we also placed several side cameras. For all cameras, we performed image cropping to focus on the area of interest and resized the images to 128 pixels by 128 pixels for the neural network to process.

Next, we collected data to train the reward classifier, which is a crucial step in defining the reward function that guides the learning process. Typically, we gathered 200 positive data points and 1000 negative data points by teleoperating the robot to perform the task. This is approximately equivalent to 10 human trajectories, assuming that each trajectory takes about 10 s. Using our data collection pipeline, as detailed in the supplementary code, it usually took around 5 min to collect these data points. In addition, we may have collected extra data to address any false-negative and false-positive issues with the reward classifier. The trained reward classifier generally achieved an accuracy of greater than 95% in the evaluation dataset.

We then collected 20 to 30 trajectories of human demonstrations solving the tasks and used them to initialize the offline demo replay buffer. For each task, we either scripted a robot reset motion or let the human operator manually reset the task at the beginning of each trajectory, such as the USB pick-insertion task. Last, we started the policy training process. During this phase, human interventions may have been provided to the policy if necessary, until the policy converged. We should avoid persistently providing long, sparse interventions that lead to task successes. Such an intervention strategy will cause overestimation of the value function, particularly in the early stages of the training process, which can result in unstable training dynamics. Figure S19 presents the statistics of such interventions during our experiments.

Statistical analysis

We evaluated the success rate of all experiments by calculating the proportion of successful trials relative to the total number of trials. We calculated the cycle time as the mean time taken to complete the task across the successful trials. Table 1 reports the success rate difference between IL and HIL-SERL in ($+X\%$) where $X = \frac{\text{SR}_{\text{HIL-SERL}} - \text{SR}_{\text{IL}}}{\text{SR}_{\text{HIL-SERL}}} * 100$ and the cycle time difference as ($Y\times$ faster) where $Y = \frac{\text{CT}_{\text{IL}}}{\text{CT}_{\text{HIL-SERL}}}$.

Supplementary Materials

The PDF file includes:

Supplementary Experimental Details
Figs. S1 to S19
Tables S1 to S12

Other Supplementary Material for this manuscript includes the following:

Movies S1 to S5

REFERENCES AND NOTES

1. J. Cui, J. Trinkle, Toward next-generation learned robot manipulation. *Sci. Robot.* **6**, eabd9461 (2021).
2. J. Hwangbo, J. Lee, A. Dosovitskiy, D. Bellicoso, V. Tsounis, V. Koltun, M. Hutter, Learning agile and dynamic motor skills for legged robots. *Sci. Robot.* **4**, eaau5872 (2019).
3. J. Lee, J. Hwangbo, L. Wellhausen, V. Koltun, M. Hutter, Learning quadrupedal locomotion over challenging terrain. *Sci. Robot.* **5**, eabc5986 (2020).
4. T. Chen, M. Tippur, S. Wu, V. Kumar, E. Adelson, P. Agrawal, Visual dexterity: In-hand reorientation of novel and complex object shapes. *Sci. Robot.* **8**, eadc9244 (2023).
5. A. Loquercio, E. Kaufmann, R. Ranftl, M. Müller, V. Koltun, D. Scaramuzza, Learning high-speed flight in the wild. *Sci. Robot.* **6**, eabg5810 (2021).
6. OpenAI, I. Akkaya, M. Andrychowicz, M. Chociej, M. Litwin, B. McGrew, A. Petron, Paino, M. Plappert, G. Powell, R. Ribas, J. Schneider, N. Tezak, J. Tworek, P. Welinder, L. Weng, Q. Yuan, W. Zaremba, L. Zhang, Solving Rubik's cube with a robot hand. arXiv:1910.07113 [cs.LG] (2019).
7. D. Kalashnikov, A. Irpan, P. Pastor, J. Ibarz, A. Herzog, E. Jang, D. Quillen, E. Holly, M. Kalakrishnan, V. Vanhoucke, S. Levine, QT-Opt: Scalable deep reinforcement learning for vision-based robotic manipulation. arXiv:1806.10293 [cs.LG] (2018).
8. D. Kalashnikov, J. Varley, Y. Chebotar, B. Swanson, R. Jonschkowski, C. Finn, S. Levine, K. Hausman, MT-Opt: Continuous multi-task robotic reinforcement learning at scale. arXiv:2104.08212 [cs.RO] (2021).
9. E. Theodorou, J. Buchli, S. Schaal, A generalized path integral control approach to reinforcement learning. *J. Mach. Learn. Res.* **11**, 3137–3181 (2010).
10. Y. Chebotar, M. Kalakrishnan, A. Yahya, A. Li, S. Schaal, S. Levine, Path integral guided policy search. arXiv:1610.00529 [cs.RO] (2016).
11. P. J. Ball, L. Smith, I. Kostrikov, S. Levine, Efficient online reinforcement learning with offline data. arXiv:2302.02948 [cs.LG] (2023).
12. T. Tang, H.-C. Lin, Y. Zhao,W. Chen, M. Tomizuka, "Autonomous alignment of peg and hole by force/torque measurement for robotic assembly" in *2016 IEEE International Conference on Automation Science and Engineering (CASE)* (IEEE, 2016), pp. 162–167.
13. S. Jin, X. Zhu, C. Wang, M. Tomizuka, "Contact pose identification for peg-in-hole assembly under uncertainties" in *2021 American Control Conference (ACC)* (IEEE, 2021), pp. 48–53.
14. A. S. Morgan, B. Wen, J. Liang, A. Boularias, A. M. Dollar, K. Bekris, "Vision-driven compliant manipulation for reliable, high-precision assembly tasks" in *Proceedings of Robotics: Science and Systems* (MIT Press Journals, 2021), 10.15607/RSS.2021.XVII.070.
15. J. Su, C. Liu, R. Li, Robot precision assembly combining with passive and active compliant motions. *IEEE Trans. Ind. Electron.* **69**, 8157–8167 (2022).
16. B. Tang, M. A. Lin, I. Akinola, A. Handa, G. S. Sukhatme, F. Ramos, D. Fox, Y. Narang, IndustReal: Transferring contact-rich assembly tasks from simulation to reality. arXiv:2305.17110 [cs.RO] (2023).
17. M. Noseworthy, B. Tang, B. Wen, A. Handa, C. Kessens, N. Roy, D. Fox, F. Ramos, Y. Narang, I. Akinola, FORGE: Force-guided exploration for robust contact-rich manipulation under uncertainty. *IEEE Robot. Autom. Lett.* **10**, 4436–4443 (2025).

18. Y. Narang, K. Storey, I. Akinola, M. Macklin, P. Reist, L. Wawrzyniak, Y. Guo, A. Moravanszky, G. State, M. Lu, A. Handa, D. Fox, Factory: Fast contact for robotic assembly. arXiv:2205.03532 [cs.RO] (2022).
19. Y. Guo, B. Tang, I. Akinola, D. Fox, A. Gupta, Y. Narang, SRSA: Skill retrieval and adaptation for robotic assembly tasks. arXiv:2503.04538 [cs.RO] (2025).
20. B. Tang, I. Akinola, J. Xu, B.Wen, A. Handa, K. V.Wyk, D. Fox, G. S. Sukhatme, F. Ramos, Y. Narang, AutoMate: Specialist and generalist assembly policies over diverse geometries. arXiv:2407.08028 [cs.RO] (2024).
21. O. Spector, V. Tchuiev, D. D. Castro, InsertionNet 2.0: Minimal contact multi-step insertion using multimodal multiview sensory input. arXiv:2203.01153 [cs.RO] (2022).
22. H. Chang, A. Boularias, S. Jain, "Insert-one: One-shot robust visual-force servoing for novel object insertion with 6-DoF tracking" in *2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2024)* (IEEE, 2024), pp. 2935–2942.
23. H.-C. Song, M.-C. Kim, J.-B. Song, "USB assembly strategy based on visual servoing and impedance control" in *2015 12th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)* (IEEE, 2015), pp. 114–117.
24. K. J. Astrom, R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers* (Princeton Univ. Press, 2008).
25. M. Mason, K. Lynch, "Dynamic manipulation" in *Proceedings of 1993 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS '93)*, (IEEE, 1993), pp. 152–159.
26. P. Kormushev, S. Calinon, D. G. Caldwell, "Robot motor skill coordination with EM-based reinforcement learning" in *2010 IEEE/RSJ International Conference on Intelligent Robots and Systems* (IEEE, 2010), pp. 3232–3237.
27. A. J. Ijspeert, J. Nakanishi, H. Hoffmann, P. Pastor, S. Schaal, Dynamical movement primitives: Learning attractor models for motor behaviors. *Neural Comput.* **25**, 328–373 (2013).
28. N. Fazeli, M. Oller, J. Wu, Z. Wu, J. B. Tenenbaum, A. Rodriguez, See, feel, act: Hierarchical learning for complex manipulation skills with multisensory fusion. *Sci. Robot.* **4**, eaav3123 (2019).
29. J. Luo, C. Xu, X. Geng, G. Feng, K. Fang, L. Tan, S. Schaal, S. Levine, Multistage cable routing through hierarchical imitation learning. *IEEE Trans. Robot.* **40**, 1476–1491 (2024).
30. S. Jin, C. Wang, M. Tomizuka, "Robust deformation model approximation for robotic cable manipulation" in *2019 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2019), pp. 6586–6593.
31. V. Viswanath, K. Shivakumar, J. Ajmera, M. Parulekar, J. Kerr, J. Ichnowski, R. Cheng, T. Kollar, K. Goldberg, HANDLOOM: Learned tracing of one-dimensional objects for inspection and manipulation. arXiv:2303.08975 [cs.RO] (2023).
32. K. Shivakumar, V. Viswanath, A. Gu, Y. Avigal, J. Kerr, J. Ichnowski, R. Cheng, T. Kollar, K. Goldberg, "SGTM 2.0: Autonomously untangling long cables using interactive perception" in *2023 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2023), pp. 5837–5843.
33. V. Viswanath, K. Shivakumar, J. Kerr, B. Thananjeyan, E. Novoseller, J. Ichnowski, A. Escontrela, M. Laskey, J. E. Gonzalez, K. Goldberg, Autonomously untangling long cables. arXiv:2207.07813 [cs.RO] (2022).
34. M. Riedmiller, T. Gabel, R. Hafner, S. Lange, Reinforcement learning for robot soccer. *Auton. Robots* **27**, 55–73 (2009).
35. S. Levine, C. Finn, T. Darrell, P. Abbeel, End-to-end training of deep visuomotor policies. *J. Mach. Learn. Res.* **17**, 1–40 (2016).
36. J. Luo, O. Sushkov, R. Pevceviciute, W. Lian, C. Su, M. Vecerik, N. Ye, S. Schaal, J. Scholz, "Robust multi-modal policies for industrial assembly via reinforcement learning and demonstrations: A large-scale study" in *Proceedings of Robotics: Science and Systems* (MIT Press Journals, 2021), 10.15607/RSS.2021.XVII.088.
37. Y. Yang, K. Caluwaerts, A. Iscen, T. Zhang, J. Tan, V. Sindhwani, "Data efficient reinforcement learning for legged robots" in *Conference on Robot Learning (PMLR)* (MLResearchPress, 2020), pp. 1–10.
38. A. Zhan, R. Zhao, L. Pinto, P. Abbeel, M. Laskin, "A framework for efficient robotic manipulation" in *DEEP RL Workshop NeurIPS 2021* (2021), pp. 1–15.
39. J. Tebbe, L. Krauch, Y. Gao, A. Zell, "Sample-efficient reinforcement learning in robotic table tennis" in *2021 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2021), pp. 4171–4178.
40. I. Popov, N. Heess, T. Lillicrap, R. Hafner, G. Barth-Maron, M. Vecerik, T. Lampe, Y. Tassa, T. Erez, M. Riedmiller, Data-efficient deep reinforcement learning for dexterous manipulation. arXiv:1704.03073 [cs.LG] (2017).
41. J. Luo, E. Solowjow, C. Wen, J. A. Ojea, A. M. Agogino, A. Tamar, P. Abbeel, "Reinforcement learning on variable impedance controller for high-precision robotic assembly" in *2019 International Conference on Robotics and Automation (ICRA)* (IEEE, 2019), pp. 3080–3087.
42. T. Z. Zhao, J. Luo, O. Sushkov, R. Pevceviciute, N. Heess, J. Scholz, S. Schaal, S. Levine, "Offline meta-reinforcement learning for industrial insertion" in *2022 International Conference on Robotics and Automation (ICRA)* (IEEE, 2022), pp. 6386–6393.
43. Z. Hu, A. Rovinsky, J. Luo, V. Kumar, A. Gupta, S. Levine, REBOOT: Reuse data for bootstrapping efficient real-world dexterous manipulation. arXiv:2309.03322 [cs.LG] (2024).
44. T. Johannink, S. Bahl, A. Nair, J. Luo, A. Kumar, M. Loskyll, J. A. Ojea, E. Solowjow, S. Levine, "Residual reinforcement learning for robot control" in *2019 International Conference on Robotics and Automation (ICRA)* (IEEE, 2019), pp. 6023–6029.
45. H. Hu, S. Mirchandani, D. Sadigh, Imitation bootstrapped reinforcement learning. arXiv:2311.02198 [cs.LG] (2024).
46. A. Rajeswaran, V. Kumar, A. Gupta, G. Vezzani, J. Schulman, E. Todorov, S. Levine, "Learning complex dexterous manipulation with deep reinforcement learning and demonstrations" in *Proceedings of Robotics: Science and Systems* (MIT Press Journals, 2018), 10.15607/RSS.2018.XIV.049.
47. G. Schoettler, A. Nair, J. Luo, S. Bahl, J. Aparicio Ojea, E. Solowjow, S. Levine, "Deep reinforcement learning for industrial insertion tasks with visual inputs and natural rewards" in *2020 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2020) pp. 5548–5555.
48. J. Luo, Z. Hu, C. Xu, Y. L. Tan, J. Berg, A. Sharma, S. Schaal, C. Finn, A. Gupta, S. Levine, "SERL: A software suite for sample-efficient robotic reinforcement learning" in *2024 IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2024), pp. 16961–16969.
49. I. Kostrikov, L. M. Smith, S. Levine, "Demonstrating a walk in the park: Learning to walk in 20 minutes with model-free reinforcement learning" in *Proceedings of Robotics: Science and Systems* (MIT Press Journals, 2023), 10.15607/RSS.2023.XIX.056.
50. J. Luo, P. Dong, Y. Zhai, Y. Ma, S. Levine, RLIF: Interactive imitation learning as reinforcement learning. arXiv:2311.12996 [cs.AI] (2023).
51. T. Hester, P. Stone, Texplore: Real-time sample-efficient reinforcement learning for robots. *Mach. Learn.* **90**, 385–429 (2012).
52. P. Wu, A. Escontrela, D. Hafner, P. Abbeel, K. Goldberg, "DayDreamer: World models for physical robot learning" in *Proceedings of the 6th Conference on Robot Learning* (MLResearchPress, 2023), pp. 2226–2240.
53. A. Nagabandi, K. Konolige, S. Levine, V. Kumar, "Deep dynamics models for learning dexterous manipulation" in *Proceedings of the Conference on Robot Learning* (MLResearchPress, 2020), pp. 1101–1112.
54. R. Rafailov, T. Yu, A. Rajeswaran, C. Finn, "Offline reinforcement learning from images with latent space models" in *Proceedings of the 3rd Conference on Learning for Dynamics and Control* (MLResearchPress, 2021), pp. 1154–1168.
55. J. Luo, E. Solowjow, C. Wen, J. A. Ojea, A. M. Agogino, "Deep reinforcement learning for robotic assembly of mixed deformable and rigid objects" in *2018 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)* (IEEE, 2018), pp. 2062–2069.
56. H. Zhu, A. Gupta, A. Rajeswaran, S. Levine, V. Kumar, "Dexterous manipulation with deep reinforcement learning: Efficient, general, and low-cost" in *International Conference on Robotics and Automation (ICRA)* (IEEE, 2019), pp. 3651–3657.
57. J. Fu, A. Singh, D. Ghosh, L. Yang, S. Levine, "Variational inverse control with events: A general framework for data-driven reward definition" in *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)* (Curran Associates, 2018), pp. 8538–8547.
58. K. Li, A. Gupta, A. Reddy, V. H. Pong, A. Zhou, J. Yu, S. Levine, "MURAL: Meta-learning uncertainty-aware rewards for outcome-driven reinforcement learning" in *Proceedings of the 38th International Conference on Machine Learning, ICML 2021* (MLResearchPress, 2021), pp. 6346–6356.
59. Y. Du, K. Konyushkova, M. Denil, A. Raju, J. Landon, F. Hill, N. de Freitas, S. Cabi, Vision-language models as success detectors. arXiv:2303.077280 [cs.LG] (2023).
60. P. Mahmoudieh, D. Pathak, T. Darrell, "Zero-shot reward specification via grounded natural language" in *International Conference on Machine Learning, ICML 2022* (MLResearchPress, 2022), pp. 14743–14752.
61. L. Fan, G. Wang, Y. Jiang, A. Mandlekar, Y. Yang, H. Zhu, A. Tang, D. Huang, Y. Zhu, A. Anandkumar, "MineDojo: Building open-ended embodied agents with internet-scale knowledge" in *Advances in Neural Information Processing Systems 35* (Curran Associates, 2022), pp. 18343–18362.
62. Y. J. Ma, S. Sodhani, D. Jayaraman, O. Bastani, V. Kumar, A. Zhang, "VIP: Towards universal visual reward and representation via value-implicit pre-training" in *The Eleventh International Conference on Learning Representations, ICLR 2023* (ICLR, 2023), pp. 1–35.
63. Y. J. Ma, V. Kumar, A. Zhang, O. Bastani, D. Jayaraman, "LIV: Language-image representations and rewards for robotic control" in *International Conference on Machine Learning, ICML 2023* (MLResearchPress, 2023), pp. 23301–23320.
64. A. Gupta, J. Yu, T. Z. Zhao, V. Kumar, A. Rovinsky, K. Xu, T. Devlin, S. Levine, "Reset-free reinforcement learning via multi-task learning: Learning dexterous manipulation behaviors without human intervention" in *IEEE International Conference on Robotics and Automation, ICRA 2021* (IEEE, 2021), pp. 6664–6671.
65. A. Sharma, K. Xu, N. Sardana, A. Gupta, K. Hausman, S. Levine, C. Finn, Autonomous reinforcement learning: Benchmarking and formalism. arXiv:2112.09605 [cs.LG] (2021).
66. H. Zhu, J. Yu, A. Gupta, D. Shah, K. Hartikainen, A. Singh, V. Kumar, S. Levine, "The ingredients of real world robotic reinforcement learning" in *8th International Conference on Learning Representations, ICLR 2020* (ICLR, 2020), pp. 1–21.
67. A. Xie, F. Tajwar, A. Sharma, C. Finn, "When to ask for help: Proactive interventions in autonomous reinforcement learning" in *Advances in Neural Information Processing Systems 35 (NeurIPS 2022)* (Curran Associates Inc., 2022), pp. 16918–16930.

68. A. Sharma, A. M. Ahmed, R. Ahmad, C. Finn, Self-improving robots: End-to-end autonomous visuomotor reinforcement learning. arXiv:2303.01488 [cs.RO] (2023).
69. K. Kimble, K. Van Wyk, J. Falco, E. Messina, Y. Sun, M. Shibata, W. Uemura, Y. Yokokohji, Benchmarking protocols for evaluating small parts robotic assembly systems. *IEEE Robot. Autom. Lett.* **5**, 883–889 (2020).
70. S. Ross, G. Gordon, D. 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* (2011), pp. 627–635.
71. M. Kelly, C. Sidrane, K. Driggs-Campbell, M. J. Kochenderfer, "HG-DAgger: Interactive imitation learning with human experts" in *2019 International Conference on Robotics and Automation (ICRA)* (IEEE, 2019), pp. 8077–8083.
72. C. Chi, Z. Xu, S. Feng, E. Cousineau, Y. Du, B. Burchfiel, R. Tedrake, S. Song, Diffusion policy: Visuomotor policy learning via action diffusion. arXiv:2303.04137 [cs.RO] (2024).
73. V. A. Papavassiliou, S. Russell, "Convergence of reinforcement learning with general function approximators" in *Proceedings of the 16th International Joint Conference on Artificial Intelligence* (Morgan Kaufmann Publishers Inc., 1999), pp. 748–755.
74. J. Bhandari, D. Russo, R. Singal, "A finite time analysis of temporal dilerence learning with linear function approximation" in *Proceedings of the 31st Conference On Learning Theory* (MLResearchPress, 2018), pp. 1691–1692.
75. C. Jin, Z. Yang, Z. Wang, M. I. Jordan, "Provably efficient reinforcement learning with linear function approximation" in *Proceedings of Thirty Third Conference on Learning Theory* (MLResearchPress, 2020), pp. 2137–2143.
76. L. F. Yang, M. Wang, Sample-optimal parametric Q-learning using linearly additive features. arXiv:1902.04779 [cs.LG] (2019).
77. R. R. Burridge, A. A. Rizzi, D. E. Koditschek, Sequential composition of dynamically dexterous robot behaviors. *Int. J. Robot. Res.* **18**, 534–555 (1999).
78. R. Tedrake, I. R. Manchester, M. Tobenkin, J. W. Roberts, LQR-trees: Feedback motion planning via sums-of-squares verification. *Int. J. Robot. Res.* **29**, 1038–1052 (2010).
79. T. Marcucci, R. Deits, M. Gabiccini, A. Bicchi, R. Tedrake, "Approximate hybrid model predictive control for multi-contact push recovery in complex environments" in *2017 IEEERAS 17th International Conference on Humanoid Robotics (Humanoids)* (2017), pp. 31–38.
80. F. R. Hogan, A. Rodriguez, Feedback control of the pusher-slider system: A story of hybrid and underactuated contact dynamics. arXiv:1611.08268 [cs.RO] (2016).
81. B. Aceituno-Cabezas, A. Rodriguez, "A global quasi-dynamic model for contact-trajectory optimization" in *Proceedings of Robotics: Science and Systems* (RSS Foundation, 2020), 10.15607/RSS.2020.XVI.047.
82. J. Jina, A. K. Bhattacharya, A. Walton, Applying lean principles for high product variety and low volumes: Some issues and propositions. *Logist. Inf. Manag.* **10**, 5–13 (1997).
83. R. Shah, P. T. Ward, Lean manufacturing: Context, practice bundles, and performance. *J. Oper. Manag.* **21**, 129–149 (2003).
84. Z. L. Gan, S. N. Musa, H. J. Yap, A review of the high-mix, low-volume manufacturing industry. *Appl. Sci.* **13**, 1687 (2023).
85. A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, J. Dabis, C. Finn, K. Gopalakrishnan, K. Hausman, A. Herzog, J. Hsu, J. Ibarz, B. Ichter, A. Irpan, T. Jackson, S. Jesmonth, N. J. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, K.-H. Lee, S. Levine, Y. Lu, U. Malla, D. Manjunath, I. Mordatch, O. Nachum, C. Parada, J. Peralta, E. Perez, K. Pertsch, J. Quiambao, K. Rao, M. Ryoo, G. Salazar, P. Sanketi, K. Sayed, J. Singh, S. Sontakke, A. Stone, C. Tan, H. Tran, V. Vanhoucke, S. Vega, Q. Vuong, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, B. Zitkovich, RT-1: Robotics transformer for real-world control at scale. arXiv:2212.06817 (2023).
86. A. Brohan, N. Brown, J. Carbajal, Y. Chebotar, X. Chen, K. Choromanski, T. Ding, D. Driess, A. Dubey, C. Finn, P. Florence, C. Fu, M. G. Arenas, K. Gopalakrishnan, K. Han, K. Hausman, A. Herzog, J. Hsu, B. Ichter, A. Irpan, N. Joshi, R. Julian, D. Kalashnikov, Y. Kuang, I. Leal, L. Lee, T.-W. E. Lee, S. Levine, Y. Lu, H. Michalewski, I. Mordatch, K. Pertsch, K. Rao, K. Reymann, M. Ryoo, G. Salazar, P. Sanketi, P. Sermanet, J. Singh, A. Singh, R. Soricut, H. Tran, V. Vanhoucke, Q. Vuong, A. Wahid, S. Welker, P. Wohlhart, J. Wu, F. Xia, T. Xiao, P. Xu, S. Xu, T. Yu, B. Zitkovich, RT-2: Vision-language-action models transfer web knowledge to robotic control. arXiv:2307.15818 [cs.RO] (2023).
87. Open X-Embodiment Collaboration, A. Padalkar, A. Pooley, A. Jain, A. Bewley, A. Herzog, A. Irpan, A. Khazatsky, A. Rai, A. Singh, A. Brohan, A. Raffin, A.Wahid, B. Burgess-Limerick, B. Kim, B. Sch¨olkopf, B. Ichter, C. Lu, C.Xu, C. Finn, C.Xu, C. Chi, C. Huang, C. Chan, C. Pan, C. Fu, C. Devin, D. Driess, D. Pathak, D. Shah, D. B¨uchler, D. Kalashnikov, D. Sadigh, E. Johns, F. Ceola, F. Xia, F. Stulp, G. Zhou, G. S. Sukhatme, G. Salhotra, G. Yan, G. Schiavi, H. Su, H.-S. Fang, H. Shi, H. B. Amor, H. I. Christensen, H. Furuta, H.Walke, H. Fang, I. Mordatch, I. Radosavovic, I. Leal, J. Liang, J. Kim, J. Schneider, J. Hsu, J. Bohg, J. Bingham, J. Wu, J. Wu, J. Luo, J. Gu, J. Tan, J. Oh, J. Malik, J. Tompson, J. Yang, J. J. Lim, J. Silverio, J. Han, K. Rao, K. Pertsch, K. Hausman, K. Go, K. Gopalakrishnan, K. Goldberg, K. Byrne, K. Oslund, K. Kawaharazuka, K. Zhang, K.Majd, K. Rana, K. Srinivasan, L. Y. Chen, L. Pinto, L. Tan, L. Ott, L. Lee, M. Tomizuka, M. Du, M. Ahn, M. Zhang, M. Ding, M. K. Srirama, M. Sharma, M. J. Kim, N. Kanazawa, N. Hansen, N. Heess, N. J. Joshi, N. Suenderhauf, N. D. Palo, N. M. M. Shafiullah, O. Mees, O. Kroemer, P. R. Sanketi, P. Wohlhart, P. Xu, P. Sermanet, P. Sundaresan, Q. Vuong, R. Rafailov, R. Tian, R. Doshi, R. Martín-Martín, R. Mendonca, R. Shah, R. Hoque, R. Julian, S. Bustamante, S. Kirmani, S. Levine, S. Moore, S. Bahl, S. Dass, S. Song, S. Xu, S. Haldar, S. Adebola, S. Guist, S. Nasiriany, S. Schaal, S. Welker, S. Tian, S. Dasari, S. Belkhale, T. Osa, T. Harada, T. Matsushima, T. Xiao, T. Yu, T. Ding, T. Davchev, T. Z. Zhao, T. Armstrong, T. Darrell, V. Jain, V. Vanhoucke, W. Zhan, W. Zhou, W. Burgard, X. Chen, X. Wang, X. Zhu, X. Li, Y. Lu, Y. Chebotar, Y. Zhou, Y. Zhu, Y. Xu, Y. Wang, Y. Bisk, Y. Cho, Y. Lee, Y. Cui, Y. Wu, Y. Tang, Y. Zhu, Y. Li, Y. Iwasawa, Y. Matsuo, Z. Xu, Z. J. Cui, "Open X-embodiment: Robotic learning datasets and RT-X models" in *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)* (IEEE, 2024), pp. 6892–6903.
88. Octo Model Team, D. Ghosh, H. Walke, K. Pertsch, K. Black, O. Mees, S. Dasari, J. Hejna, T. Kreiman, C. Xu, J. Luo, Y. L. Tan, L. Y. Chen, P. Sanketi, Q. Vuong, T. Xiao, D. Sadigh, C. Finn, S. Levine, Octo: An open-source generalist robot policy. arXiv:2405.12213 [cs.RO] (2024).
89. M. J. Kim, K. Pertsch, S. Karamcheti, T. Xiao, A. Balakrishna, S. Nair, R. Rafailov, E. Foster, G. Lam, P. Sanketi, Q. Vuong, T. Kollar, B. Burchfiel, R. Tedrake, D. Sadigh, S. Levine, P. Liang, C. Finn, OpenVLA: An open-source vision-language-action model. arXiv:2406.09246 [cs.RO] (2024).
90. Y. Song, Y. Zhou, A. Sekhari, D. Bagnell, A. Krishnamurthy, W. Sun, "Hybrid RL: Using both offline and online data can make RL efficient," poster presented at the 11th International Conference on Learning Representations, Kigali, Rwanda, 1 May to 5 May 2023.
91. V. Mnih, K. Kavukcuoglu, D. Silver, A. Graves, I. Antonoglou, D. Wierstra, M. Riedmiller, Playing Atari with deep reinforcement learning. arXiv:1312.5602 [cs.LG] (2013).
92. T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, "Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor" in *Proceedings of the 35th International Conference on Machine Learning* (MLResearchPress, 2018), pp. 1861–1870.
93. T. Haarnoja, A. Zhou, P. Abbeel, S. Levine, Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. arXiv:1801.01290 [cs.LG] (2018).
94. A. Radford, J. W. Kim, C. Hallacy, A. Ramesh, G. Goh, S. Agarwal, G. Sastry, A. Askell, P. Mishkin, J. Clark, G. Krueger, I. Sutskever, Learning transferable visual models from natural language supervision. arXiv:2103.00020 [cs.CV] (2021).
95. A. Dosovitskiy, L. Beyer, A. Kolesnikov, D. Weissenborn, X. Zhai, T. Unterthiner, M. Dehghani, M. Minderer, G. Heigold, S. Gelly, J. Uszkoreit, N. Houlsby, An image is worth 16x16 words: Transformers for image recognition at scale. arXiv:2010.11929 [cs.CV] (2021).
96. A. Kolesnikov, L. Beyer, X. Zhai, J. Puigcerver, J. Yung, S. Gelly, N. Houlsby, Big transfer (BiT): General visual representation learning. arXiv:1912.11370 [cs.CV] (2020).
97. S. S. Du, S. M. Kakade, R.Wang, L. F. Yang, Is a good representation sufficient for sample efficient reinforcement learning? arXiv:1910.03016 [cs.LG] (2020).
98. K. He, X. Zhang, S. Ren, J. Sun, Deep residual learning for image recognition. arXiv:1512.03385 [cs.CV] (2023).
99. J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, L. Fei-Fei, "ImageNet: A large-scale hierarchical image database" in *2009 IEEE Conference on Computer Vision and Pattern Recognition* (IEEE, 2009), pp. 248–255.
100. A. Y. Ng, D. Harada, S. J. Russell, "Policy invariance under reward transformations: Theory and application reward shaping" in *Proceedings of the Sixteenth International Conference on Machine Learning*, *ICML '99* (Morgan Kaufmann Publishers Inc., 1999), p. 278–287.
101. C. Florensa, D. Held, X. Geng, P. Abbeel, Automatic goal generation for reinforcement learning agents. arXiv:1705.06366 [cs.LG] (2018).
102. C. Florensa, D. Held, M.Wulfmeier, M. Zhang, P. Abbeel, "Reverse curriculum generation for reinforcement learning" in *Proceedings of the 1st Annual Conference on Robot Learning* (MLResearchPress, 2017), pp. 482–495.
103. H. van Hasselt, A. Guez, D. Silver, Deep reinforcement learning with double Q-learning. arXiv:1509.06461 [cs.LG] (2015).
104. C. Jin, Z. Allen-Zhu, S. Bubeck, M. Jordan, "Is Q-learning provably efficient?" in *Advances in Neural Information Processing Systems 31 (NeurIPS 2018)* (Curran Associates, 2018), pp. 4863–4873.
105. M. G. Azar, R. Munos, B. Kappen, On the sample complexity of reinforcement learning with a generative model. arXiv:1206.6461 [cs.LG] (2012).
106. M. J. Kearns, S. P. Singh, "Near-optimal reinforcement learning in polynominal time" in *Proceedings of the Fifteenth International Conference on Machine Learning* (Morgan Kaufmann Publishers Inc., 1998), pp. 260–268.

Acknowledgments: We would like to thank K. Stachowicz and Q. Li for very helpful discussions. **Funding:** This research was partly supported by the Office of Naval Research N00014-20-1-2383 and N00014-19-12042, National Science Foundation IIS-2150826, Berkeley DeepDrive, and the AI Institute. **Author contributions:** J.L. designed the research, conceived the idea of and devised the main method in the paper, implemented the prototype, performed initial experiments, provided hands-on guidance on all experiments, analyzed and interpreted the results, led the project, and wrote and positioned the paper. C.X. contributed to the research design, maintained the main research code base, prepared experiment hardware, performed a number of experiments, and prepared figures and videos for the paper. J.W. contributed to the research design, maintained the main research code base, performed a number of experiments, and cleaned up the code base for public release. S.L. contributed to the research design,

advised the project, and edited and positioned the paper. **Competing interests:** S.L. is also affiliated with Physical Intelligence. The other authors declare that they have no competing interests. **Data and materials availability:** Accompanying code can be found at https://zenodo.org/records/16064289. Additional material can be found at https://hil-serl.github.io/.

Submitted 1 November 2024
Accepted 23 July 2025
Published 20 August 2025
10.1126/scirobotics.ads5033

Science Robotics

Precise and dexterous robotic manipulation via human-in-the-loop reinforcement learning

Jianlan Luo, Charles Xu, Jeffrey Wu, and Sergey Levine

Sci. Robot. **10** (105), eads5033. DOI: 10.1126/scirobotics.ads5033

Science Robotics (ISSN 2470-9476) is published by the American Association for the Advancement of Science. 1200 New York Avenue NW, Washington, DC 20005. The title *Science Robotics* is a registered trademark of AAAS.